**Documentación del Simulador de Evacuación Urbana**

**1. Introducción**

Este documento presenta el **Simulador de Evacuación Urbana**, una herramienta diseñada para **evaluar el desempeño de evacuación de personas en entornos urbanos**. Su propósito es estimar el **RSET (Tiempo Requerido para Evacuar)** y el **número total de personas evacuadas** tras un desastre. El modelo busca responder a la pregunta clave: **¿en cuánto tiempo se logra evacuar a la totalidad de una población, considerando parámetros como el nivel de pánico, y cuál es la ventana de tiempo suficiente para alertar y evacuar a todos de forma segura?**

Desarrollado en **Repast Simphony 2.11** con **Java 17**, el simulador permite explorar dinámicas de movimiento, congestión y eficiencia de rutas, ofreciendo una plataforma configurable para la experimentación de diferentes escenarios y parámetros urbanos predefinidos.

**2. Diseño y Arquitectura General**

El simulador se basa en un enfoque de **Modelado Basado en Agentes (ABM)**. Los componentes principales incluyen:

* **Agentes (Personas):** Representan a los individuos en el entorno. Cada agente sigue un **diagrama de estados** que incluye las fases **Pre-movimiento**, **Movimiento** y **Evacuado**. Poseen atributos como agentID, ageGroup, citizenType, state, currLocation, currSpeed, currDirection, panicLevel, helpfulnessLevel, needsAssistance, entre otros, como se detalla en la Tabla 1: "Entidades y variables de estado".
* **Entorno Urbano (Celda / Grilla):** Se modela como una **grilla bidimensional regular** (MapCellGrid) de celdas (MapCell), cada una conteniendo información sobre su ubicación, elevación y transitabilidad (isPassable). Esta grilla es la base sobre la cual se definen las áreas urbanas, incluyendo carreteras.
* **Zonas de Interés:**
  + **Zonas de Origen (initialAreas):** Áreas desde donde las personas inician la evacuación, definidas mediante un archivo espacial.
  + **Zonas de Destino/Seguras (safeAreas):** Puntos a los que los agentes intentan llegar (targetLocation) durante la evacuación, también definidas espacialmente.
  + **Carreteras/Vías de Tránsito:** Definidas espacialmente para marcar las celdas por donde los agentes pueden moverse.
* **Interacciones:** Los agentes interactúan con el entorno (navegando por las celdas transitables de la grilla, que incluyen las carreteras) y entre sí (evitando colisiones, ajustando su comportamiento emocional, y ofreciendo ayuda).

**3. Implementación Técnica**

El simulador ha sido implementado utilizando el *toolkit* de simulación **Repast Simphony 2.11** y **Java 17**.

* **Estructura del Proyecto:** El proyecto se organiza en paquetes Java que encapsulan la lógica del agente (GisAgent), la construcción del contexto del modelo (ContextCreator), y la definición de parámetros.
* **Espacios de Repast:**
  + **Continuous Space (Geography):** Utilizado para manejar la posición espacial continua de los agentes y las zonas en coordenadas geográficas (latitud, longitud), permitiendo la integración de datos geoespaciales.
  + **Grid Space (MapCellGrid, AgentGrid):** Se utilizan dos grillas discretas. MapCellGrid representa el terreno, donde cada celda (MapCell) almacena propiedades como transitabilidad (definida por zonas y carreteras). AgentGrid maneja las posiciones discretas de los agentes, permitiendo el cálculo de rutas y la detección de vecinos de forma eficiente.
* **Integración Geoespacial:**
  + El modelo carga **Shapefiles (.shp)** (procesados previamente desde **GeoJSON.io** y **QGIS**) para definir las initialAreas, safeAreas y roads.
  + Se implementa un mapeo entre coordenadas geográficas (latitud/longitud) y las coordenadas de la grilla (GridPoint) para permitir la interacción entre los datos espaciales continuos y la discretización del entorno.
* **Algoritmo de Ruta:** Los agentes calculan sus rutas hacia las zonas seguras utilizando el algoritmo **A\*** implementado directamente en el GisAgent, considerando la transitabilidad de las celdas en la MapCellGrid.
* **Configuración del Modelo:** El modelo es altamente configurable a través de **Parámetros de Modelo** expuestos en la interfaz gráfica de Repast Simphony, y a través de la carga de archivos espaciales externos.

**4. Parámetros de Configuración del Modelo**

La flexibilidad del simulador se logra a través de los siguientes parámetros, accesibles en la interfaz de Repast Simphony antes de la ejecución, que corresponden a los definidos en la Tabla 3: "Parámetros de la simulación":

* **Tiempo de Simulación Total (T\_sim):** Define la duración máxima de la simulación en segundos.
  + *Tipo:* Real
  + *Rango Recomendado:* Mayor a 0
* **Cantidad de Agentes a Simular (totalAgents):** Establece el número total de personas que participarán en la evacuación.
  + *Tipo:* Entero
  + *Rango Recomendado:* Desde 1 hasta la máxima capacidad de la población residente del escenario.
* **Radio de Cobertura de un Agente (covRadius):** Define la distancia en metros dentro de la cual un agente puede percibir a otros agentes y condiciones del entorno.
  + *Tipo:* Real
  + *Rango Recomendado:* Mayor a 0
* **Ruta de Archivo de Zonas Iniciales (initialAreasFilePath):** Ruta relativa o absoluta al archivo Shapefile (.shp) que delimita las áreas donde los agentes son distribuidos aleatoriamente al comienzo.
  + *Tipo:* Cadena de texto (string)
  + *Ejemplo de ruta:* data/POLYGON.shp
* **Ruta de Archivo de Zonas Seguras (safeAreasFilePath):** Ruta relativa o absoluta al archivo Shapefile (.shp) que contiene la definición de las áreas designadas como seguras para la evacuación.
  + *Tipo:* Cadena de texto (string)
  + *Ejemplo de ruta:* data/Zones2.shp
* **Ruta de Archivo de Carreteras (roadsFilePath):** Ruta relativa o absoluta al archivo Shapefile (.shp) que define la red de carreteras del entorno.
  + *Tipo:* Cadena de texto (string)
  + *Ejemplo de ruta:* data/roads.shp

**5. Requisitos del Sistema y Ejecución**

* **Software Requerido:**
  + Java Runtime Environment (JRE) versión 17 (preferiblemente LTS).
  + Un archivo portable del simulador (.zip) generado desde Repast Simphony 2.11.
* **Herramientas de Preparación de Datos:**
  + **GeoJSON.io:** Para la creación o edición inicial de geometrías.
  + **QGIS:** Para el procesamiento y exportación de datos geoespaciales a formato Shapefile (.shp).
  + **Scripts Propios:** Para cualquier pre-procesamiento o conversión de datos adicional.
* **Ejecución:**
  1. Descomprima el archivo .zip del simulador en una carpeta local.
  2. Navegue a la carpeta descomprimida.
  3. Ejecute el script start\_model.bat (Windows) o start\_model.command (macOS/Linux).
  4. Una vez cargada la interfaz de Repast Simphony, ajuste los parámetros deseados en la pestaña "Parameters". Asegúrese de que las rutas a los archivos Shapefile sean correctas.
  5. Presione el botón "Initialize" y luego "Run" para iniciar la simulación.

**6. Submodelos y Flujo de Simulación**

La simulación procede a través de una secuencia de submodelos que guían el comportamiento de los agentes:

* **Pre-movimiento:** Cada agente experimenta una demora inicial aleatoria antes de decidir iniciar la evacuación, representando la toma de decisiones inicial.
* **Ciclo de Decisión y Movimiento (Iterativo):** Una vez que el agente decide evacuar, entra en un ciclo repetitivo en cada paso de tiempo:
  + **Decisión de Ruta:** Los agentes determinan su trayectoria hacia una zona segura asignada. Un bajo nivel de pánico impulsa la selección de una **ruta óptima** (calculada por el algoritmo A\*), mientras que un alto nivel de pánico puede llevar a la elección de una **ruta no óptima o errática**.
  + **Análisis de Vecindad:** El agente percibe su entorno inmediato (dentro de su covRadius), identificando a otros agentes y las condiciones de la celda actual (isPassable).
  + **Ajuste del Comportamiento Emocional:** La información de la vecindad (ej. contacto visual con agentes en pánico, proximidad a zonas de peligro) influye en el panicLevel del agente.
  + **Evitar Colisiones:** Utilizando un modelo de fuerza social, el agente ajusta su dirección y velocidad (currSpeed) para prevenir colisiones con otros agentes y obstáculos, calculando así su movimiento real para el instante.
  + **Actualización de Posición:** El agente actualiza su currLocation en el entorno.
* **Transición a "Evacuado":** El ciclo termina cuando un agente alcanza una zona segura, pasando al estado Evacuado, registrando su evacTime y actualizando el evacAgents global.
* **Resultados Finales:** Al finalizar la simulación, se extrae el número total de evacuados y no evacuados, y se analiza el tiempo promedio de evacuación, registrando las variables de estado en formatos como texto plano o CSV para análisis posterior.

**7. Limitaciones y Futuras Mejoras**

Esta implementación es una prueba de concepto y presenta las siguientes limitaciones, que representan claras oportunidades de mejora futura:

* **Definición Fija del Marco Geográfico:** Las dimensiones de la grilla (GRID\_WIDTH, GRID\_HEIGHT) y los límites geográficos (MIN\_LONGITUDE, etc.) están codificados. Futuras versiones podrían permitir la definición dinámica de estos parámetros a través de la interfaz de Repast o archivos de configuración.
* **Simplificaciones del Comportamiento del Agente:** El modelo actual utiliza un comportamiento de movimiento y decisión basado en pánico. Futuras mejoras podrían incluir:
  + Modelos de evitación de obstáculos más sofisticados y dinámicos.
  + Consideración de fatiga o resistencia.
  + Modelos de toma de decisiones más complejos (ej. conocimiento de rutas seguras, búsqueda de información).
  + Interacciones sociales más ricas (ej. comunicación explícita, formación de grupos).
* **Representación de Obstáculos y Zonas de Peligro:** Actualmente, las zonas de peligro (hazardAreas) no se están cargando directamente en el ContextCreator como Shapefiles independientes. El modelo asume que las celdas **no transitables** son aquellas que no son TYPE\_INITIAL\_ZONE, TYPE\_SAFE\_ZONE o TYPE\_ROAD. Una mejora importante sería incorporar la carga explícita de hazardAreas para marcar celdas como intransitables, permitiendo la simulación de rutas bloqueadas por desastres.
* **Carga del Entorno y Datos Geoespaciales:** Aunque se utilizan Shapefiles, la complejidad del entorno urbano real (edificios, diferentes tipos de calles, obstáculos dinámicos) podría ser mejorada. La dependencia de pre-procesamiento en QGIS y GeoJSON.io es funcional, pero podría explorarse una integración más directa o formatos de datos más ricos.
* **Tipos de Desastre:** Actualmente, el desastre se representa de forma estática como zonas intransitables. Un modelo más avanzado podría simular la propagación dinámica de un desastre (ej. inundaciones, incendios).
* **Visualización y Análisis:** La visualización es funcional, pero podría mejorarse con herramientas interactivas más avanzadas o integraciones con sistemas GIS para un análisis espacial más profundo de los resultados.

\section{Preparación y Carga de Datos Geoespaciales}

La fase inicial de configuración del entorno de simulación se centra en la preparación de datos geoespaciales críticos que definen el área de estudio y su infraestructura. Este proceso comienza con la delimitación de las zonas de interés la zona inicial de evacuación y las zonas seguras. Estas geometrías son creadas como polígonos utilizando una herramienta de edición GeoJSON basada en la web como GeoJSON.io. Durante esta creación cada polígono es enriquecido con un atributo zoneType el cual clasifica la entidad como initial o safe según corresponda.

Una vez definidos estos polígonos un script de Python personalizado procesa el archivo GeoJSON resultante. Este script ejecuta varias funciones clave de automatización. Primero filtra los polígonos de entrada y los exporta a archivos GeoJSON separados designados para las zonas iniciales y las zonas seguras. Adicionalmente para la extracción de infraestructura vial calcula un bounding box que abarca la extensión geográfica combinada de todas las zonas de interés. Utilizando este bounding box el script interactúa con los datos de OpenStreetMap ya sea descargándolos directamente o recortando un archivo PBF existente mediante wget y osmconvert. Posteriormente emplea osmfilter para extraer específicamente las vías etiquetadas como highway= que representan la red de carreteras.

Finalmente este mismo script de Python es responsable de la conversión automatizada de los archivos GeoJSON generados para zonas iniciales zonas seguras y carreteras a formato Shapefile utilizando la librería geopandas que integra las capacidades de GDAL OGR. Esta automatización elimina la necesidad de herramientas GIS intermedias para la conversión. Los Shapefiles resultantes son ubicados en la carpeta /data del proyecto de Repast Simphony asegurando su carga y procesamiento directo por el ContextCreator del simulador durante la inicialización. Este flujo de trabajo garantiza una preparación de datos precisa y eficiente fundamental para la fidelidad del modelo.

Falta:

* Manejo de colisiones
* Parámetros bien definidos
* Atributos de agentes y grillas
* Caminos (A\*)
* Pasar los mapas por parámetro y las latitudes
* Verificar los algoritmos de conversión, etc todos
* Arreglar el interfaz
* Saber cómo dar output del tiempo, gráficos, videos, etc
* Automatizar todo