Problema 1 de 4
SID 2223q2

Marzo/Abril 2023

Usando la plantilla de proyecto de Dedale, implementad dos agentes A y B. Los dos agentes comienzan en posiciones aleatorias del mapa (n_A y n_B) y han de intercambiar sus posiciones, es decir, que en algún momento concreto de la ejecución de la plataforma A tiene que estar en n_B y B tiene que estar en n_A. Se han de cumplir las siguientes condiciones:

- Al inicio de la ejecución, ni A conoce n_B ni B conoce n_A.
- A y B se pueden comunicar siguiendo las reglas del entorno Dedale, usando el método *sendMessage*. El rango de comunicación de los agentes (configurado en el fichero de entidades) ha de ser 1.
- B no puede moverse hasta que tenga conocimiento de n_A.
- Puede haber más agentes en el entorno, de manera que es recomendable usar el Directory Facilitator para que A y B sepan el AID del otro. No se puede usar el DF como mecanismo de comunicación excepto para anunciar AID y tipo de agente (servicio).
- Con independencia de la cantidad de agentes que haya funcionando, A y B han de poder funcionar de manera autónoma en la plataforma suponiendo que no haya agentes que bloqueen intersecciones relevantes.

El diseño interno de los agentes deberá estar basado en alguna de las arquitecturas de agente vistas en clase de teoría. Podéis usar *behaviours* para representar los distintos componentes de las arquitecturas que seleccionéis.

No se tendrá en cuenta la eficiencia de los agentes (podéis usar cualquier método para decidir el camino a seguir) pero su comportamiento ha de garantizar que llegarán a su destino en un tiempo finito. Podéis reutilizar código de los demás agentes incluidos en la plantilla de proyecto de Dedale.

En la entrega tenéis que incluir:

- Un documento breve (.txt o .pdf) que explique:
  - Las decisiones de diseño que habéis seguido para implementar cada agente.
  - Cómo os habéis repartido las tareas entre los autores de la entrega.
- El código fuente de los dos agentes.

La evaluación se dividirá en los siguientes aspectos:

- Los agentes funcionan correctamente cualesquiera que sean las posiciones iniciales. (5/10 puntos)
- La arquitectura de cada agente está justificada y sus componentes e interacciones se reflejan en la implementación. (2/10 puntos)
- El *DirectoryFacilitator* se utiliza correctamente para el registro y descubrimiento de los agentes. (1/10 puntos)
- Los behaviours se utilizan de manera que cada agente pueda funcionar asíncronamente y sin bloqueos (por ejemplo, mediante el uso de behaviours paralelos, templates de mensajes, la adición dinámica de behaviours, etc.). (2/10 puntos)
- *Extra:* Podríamos decir que A tiene el rol de buscar y comunicar y B tiene el rol de esperar y moverse. Haced que al llegar B a su nodo de destino, se intercambien los roles, de modo que B tenga el rol de buscar y comunicar y A el de esperar y moverse, y cuando estos roles se cumplan con éxito se inviertan de nuevo, en un bucle infinito. Podéis ser creativos en la manera de conseguirlo. Documentad qué habéis tenido que hacer de diferente respecto a la versión *estándar*. (2/10 puntos)

La entrega y corrección del problema se hará en grupos de como máximo 3 personas y contará un 1/4 de la nota de problemas. Deberéis entregar vuestra solución antes del final del viernes 14 de abril en el espacio que se habilitará en el Racó.