Problema 1 de 4

SID 2223q2

Grupo: Haobin Guo, Lin Jinheng, Lin Zheshuo

Usando la plantilla del proyecto de Dedale, implementamos dos agentes A y B, de la misma clase Agente_P1.

Los dos agentes comienzan en posiciones aleatorias del mapa (nA, nB) y se intercambian sus posiciones, es decir, que en algún momento concreto de la ejecución de la plataforma A está en nB y B está en nA. Al principio A no conoce a B ni B conoce nA. B no se mueve hasta tener conocimiento de nA

Podemos decir que A tiene el rol de buscar y B tiene el rol de esperar y moverse. Hacemos que B al llegar al destino se intercambien los roles y A al llegar al destino se intercambien los roles, cuando estos roles se cumplan con éxito se invierten de nuevo, en un bucle infinito.

La clase Agente_P1 es un extends de la clase AbstractDedaleAgent, que tiene como atributos:

- **Map<String,String> agents**: Representa los agentes que están en el mapa, y su posición, dependiendo de tiempo, es decir al recibir la posición de un agente, también recibe el tiempo que va ha estar allí, lo cual en **agents** el valor de la posición solo va estar un tiempo específico.
- **String initPos**: Guarda la posición inicial del agente.
- **String dest**: Guarda la posición destino del agente, si tiene.
- **String objective**: Guarda el agente que quiere buscar.
- **ROLE role**: Indica que rol tiene, puede ser SEARCH o WAIT.
- **STATE state**: Indica que estado esta RUNNING , STOP o FINISH.
- **MapRepresentation myMap**: Estructura para guardar el mapa, procesar los caminos.

La clase MapRepresentation viene dada por la plataforma, pero no hace algunas cosas que queremos, lo cual, lo copiamos y modificamos algunas funciones.

Se usa los siguientes Behaviour:

- **InitBehaviour**: Es un WalkerBehaviour que se despierta al 1000 ms después del setup del agente, para evitar inicializar los otros behaviour en el setup, que provocaban algunos errores, por que se necesitaban cosas que aún no se había inicializado. En este behaviour, se obtiene la existencia de otros agentes con el DirectoryFacilitator, se inicializa el **myMap**, el **ExploreBehaviour**, **ShareInfoBehaviour** y el **ShareInitPosBehaviour**.

- **ExploreBehaviour**: Cada vez que se inicializa, empieza de cero a explorar el mapa. Si es el rol WAIT finaliza al momento. Si se encuentra con el **dest**, finaliza e inicializa

RestartBehaviour, o si terminado de explorar el mapa inicializa el **GoPosBehaviour.**

- **ShareInfoBehaviour**: Envía la posición actual que está con el tiempo que va estar, envía los datos del **myMap** que tiene. Recibe los datos del **myMap** de los otros agentes. Recibe la posición y el tiempo que van ha estar de los otros agentes, con la posición, actualiza **agents**, e inicializa el **UpdateAgentPosBehaviour** con el tiempo indicado.

- **ShareInitPosBehaviour**: Envía su posición inicial y recibe la posición inicial del otro agente, por un protocolo específico(`SHARE-INIT-POS`), Al recibir la posición del otro agente inicializa el **GoPosBehaviour** para el rol WAIT.

- **GoPosBehaviour**: Va a la posición indicada. Al finalizar inicializa **RestartBehaviour** a un tiempo *`RESTARTTIME`*

- **UpdateAgentPosBehaviour**: Actualiza el agente específico de **agents** con un valor "-1".

- **RestartBehaviour**: Reinicia el **initPos**, el **dest** del agente, e intercambia su rol y establece el estado correspondiente. Inicializa de nuevo ExploreBehaviour, de forma que volverá a explorar todo el mapa.

Agente_P1 es un agente reactivo con estado que tiene como sensor de percepción el **ShareInfoBehaviour**y el **ShareInitPosBehaviour**, tiene como actuador el **ExploreBehaviour** y **GoPosBehaviour**, Los otros Behaviour son para el cambio de estado.

El intercambio de roles se consigue mediante el **RestartBehaviour**, y con los atributos **role y state** de la clase Agent_P1.