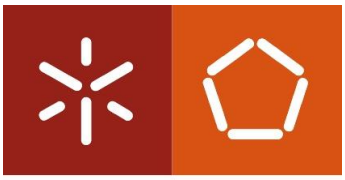


Universidade do Minho

Departamento de Informática

Trabalho Prático
Agentes Inteligentes
Grupo 3

Bruno Martins (a80410) Catarina Machado (a81047)
Filipe Monteiro (a80229)

18 de Janeiro de 2020

Conteúdo

1 Introdução

Os Sistemas Multiagentes são uma subárea da Inteligência Artificial Distribuída e concentram-se no estudo de agentes autónomos no universo multiagente. Neste sistema, três ou mais agentes interagem entre si e trabalham cooperativamente ou competitivamente para atingir um objetivo. Paralelismo, escalabilidade e a possibilidade de divisão em problemas menores são as principais características destes sistemas.

O objetivo do sistema multiagente do presente trabalho prático é monitorizar e resolver catástrofes naturais, através da aplicação de vários sensores virtuais de captura de localização GPS, representados por agentes. O cenário do nosso problema consiste num simulador de combate a incêndios florestais. Neste simulador, existirão Agentes Participativos, que são os veículos autónomos e inteligentes (voadores e terrestres), mais especificamente aeronaves, drones e camiões de bombeiros. Para além disso, existirá um Agente Central, o quartel de bombeiros, responsável pela monitorização, coordenação e distribuição de tarefas entre os agentes participativos.

Desta forma, começaremos por apresentar um estado da arte sobre os agentes e a sua aplicação em domínios concretos, abordando as diferentes propriedades, e, em seguida, explicamos como pensamos, concebemos e modelamos uma arquitetura distribuída baseada em agentes para o dado problema. Para isso, começamos por expor a Contextualização geral da nossa solução, seguido da sua Arquitetura geral, e, posteriormente, o respetivo Protocolo de Comunicação.É também abordado o Processamento Interno dos Agentes. Em cada um dos três últimos tópicos mencionados foram aplicadas metodologías *Agent UML* para formalizar os protocolos de interação entre agentes.

Após feita a introdução conceptual e modelação do sistema é descrita a implementação feita com recurso à ferramenta *JADE* para a criação, manutenção e ação dos agentes. Para que as funcionalidades ficassem visíveis foi utilizada a biblioteca *JFreeChart* para reproduzir o mundo criado num mapa e as constantes alterações no estado do mesmo. Além disso são também exibidas estatísticas relevantes acerca do estado do mundo tal como incêndios ativos no mundo.

2 Estado de Arte

2.1 Descrição do Problema

O nosso sistema multiagente consiste num simulador de combate a incêndios florestais. Neste sistema, existem três diferentes tipos de agentes: os agentes participativos, que são os veículos autónomos e inteligentes (voadores e terrestres), tais como aeronaves, drones e camiões de bombeiros; o agente central, ou seja, o quartel, que é responsável por coordenar e distribuir tarefas entre os agentes participativos. Estas decisões são influenciadas por um conjunto de fatores existentes no meio, nomeadamente as localizações de locais de abastecimento disponíveis de água e combustível. Por fim, existe um agente secundário, o ateador de fogos, que gerará incêndios em momentos/posições aleatórias.

O objetivo do sistema é monitorizar e resolver estas catástrofes naturais, de forma a que todos os fogos sejam apagados o mais rapidamente possível, colmatando assim num ambiente livre de chamas.

2.2 Revisão da Literatura

O sistema que estamos a desenvolver basear-se-á na existência de uma organização de agentes que, cooperando entre si irão resolver um problema. Neste caso, estes agentes irão "tratar" o problema por completo, mas, na sua essência, seria para ajudar na alocação de recursos, otimização de rotas, na própria gestão do meio.

Atualmente, já vários estudos foram feitos na área, onde se procura criar uma plataforma que consiga gerir, eficientemente, uma força de ajuda no caso de um desastre natural. Isto, porque o ser Humano é limitado, podendo dar resposta, da melhor forma, a um número pequeno de ocorrências comparativamente a uma máquina. Também o fazem com falta de informação do panorama geral, às vezes provocando decisões não tão acertadas. Com uma máquina, esta sabendo o estado por

completo dos recursos, das zonas de perigo, entre outros fatores, pode tomar as decisões mais lógicas possíveis, indicando simplesmente aos respetivos meios de combate o que é necessário.

Em 2010, um estudo foi feito na criação de um sistema de multiagentes capaz de auxiliar as equipas de resgate no caso de um desastre natural, no Paquistão, depois de um enorme tremor de terra, em 2006, ter tido um efeito devastador, onde as equipas de resgate se encontraram "presas" pela gravidade da situação. Esta plataforma, chamada de **FMSIND**[1] (*A Framework of Multi-Agent Systems Interaction during Natural Disaster*), consistia numa grande divisão de tarefas por variadíssimos agentes, onde no final, um último agente iria fazer a comunicação ao respetivo serviço de ajuda (bombeiros, polícias, médicos, entre outros). Arquiteturalmente consiste no seguinte:

Um agente detentor de sensores (**Sensor Agent**) fica à escuta de um desastre. Após estímulo, notifica um sistema de decisão do tipo de desastre (**DSS**) que irá avaliar a situação e enviar o que necessita para resolução do problema de volta para o agente sensorial. Este por sua vez irá enviar ao agente supervisor (**Supervisor Agent**) o que necessita e o agente supervisor irá distribuir a gestão dos diferentes tipos de recursos pelos diferentes agentes responsáveis por cada área (**On-Site Agents**). Cada um destes, depois de comunicar com a base de dados sobre os recursos disponíveis, irá comunicar com o seu respetivo agente responsável indicando a ação que é necessário tomar (**Remote-Site Agents**). Por fim, este servirá de ponte entre o sistema e os meios de suporte, alertando os meios conforme o indicado na BD.

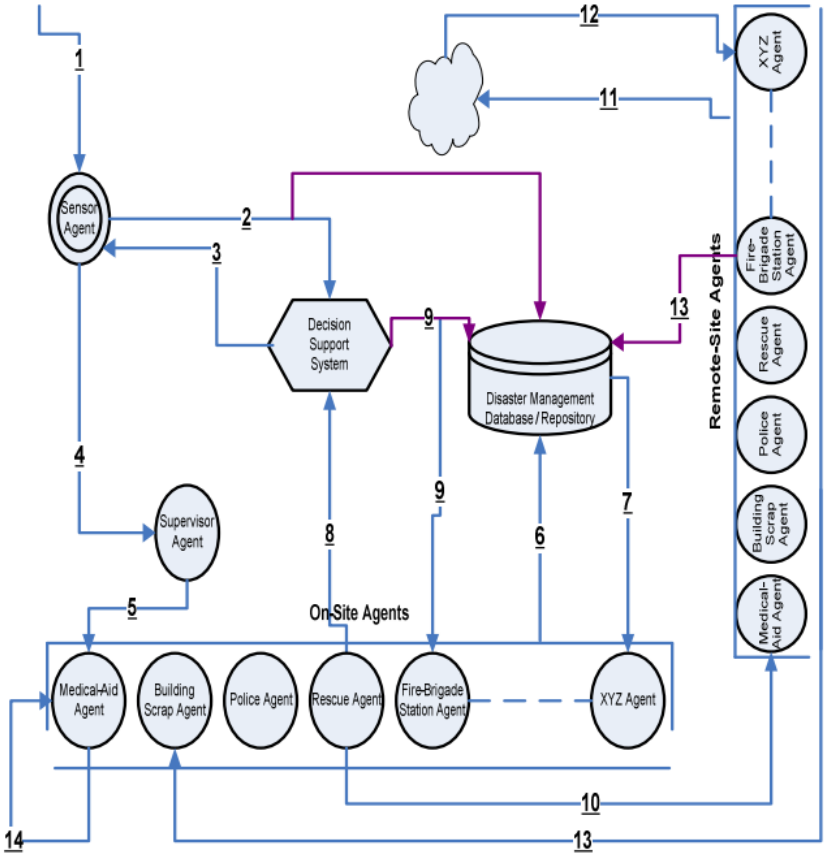


Figura 1: Arquitetura do *FMSIND*

A nossa solução e o estudo descrito anteriormente foram baseados em ferramentas já criadas, nomeadamente:

- **EQ-Rescue** [3], uma ferramenta que simula o aparecimento de tremores de terra e o tratamento deste. Este consiste de 3 módulos: Disaster World Model, que é usado para repre-

sentação do mundo, o Resource Model, representante dos recursos existentes e, por fim, o Emergency Operation Center (EOC), módulo responsável pela tomada de decisões;

- **Knowledge Oriented Sensor Web**[2], uma plataforma onde os agentes trabalham cooperativamente para, não só determinar o estado da situação mas também para tomar ações. O sistema consiste em duas partes, uma responsável por avaliar o ambiente e outra por tratar das respostas. Estas ficam ligadas através de uma base de dados e de um sistema de decisão comunicando por específicas mensagens ou através uma plataforma web.

3 Contextualização da Solução

O mapa da Floresta do simulador é constituído por **habitações** (locais onde os agentes participativos terrestres não serão capazes de "pisar"), locais de abastecimento de **água** (rios, mares, lagoas, entre outros) e locais de abastecimento de **combustível** (bombas de combustível).

No início de cada simulação é gerado um mapa, onde as habitações e os locais de abastecimento de água e de combustível serão distribuídos por posições aleatórias. Posteriormente, o Quartel irá dividir o mapa por **zonas** (delimitadas por 4 posições, por isso, terão que ser quadradas ou retangulares), sendo que cada uma deverá conter pelo menos um posto de abastecimento de água e de combustível. Em seguida, os agentes serão proporcionalmente distribuídos pelas zonas, ou seja, quanto maior for uma zona, mais agentes participativos estarão encarregues pela mesma. Para além disso, também terá em consideração os diferentes tipos de veículos existentes, de modo a que se encontrem equilibradamente distribuídos por todo o mapa. Essa posição delegada a cada um dos veículos no início da simulação será considerada a sua **posição ideal**, logo, o bombeiro deverá permanecer nessa posição até ir combater um incêndio e regressar para a mesma assim que o incêndio se extinguir.

Um bombeiro encontrar-se-á sempre num dos seguintes estados: **em repouso** quando não está encarregue de nenhum incêndio, **a caminho** para combater um incêndio, **em ação** quando está a combatê-lo e **a regressar** quando está a voltar para a sua posição ideal.

Posteriormente, o **ateador de fogos**, através de uma taxa de probabilidade aleatória e variável, irá gerar um fogo numa posição arbitrária do mapa. Nesta posição não poderão existir habitações, nem postos de abastecimento de água e combustível. Depois de iniciado um fogo, este poderá **expandir-se** pois estará associado a uma taxa de expansão, que variará de acordo com o tempo de duração do incêndio e com o ambiente circundante (quanto mais húmido, ou seja, com abastecimentos de água, menos probabilidade terá de se expandir, e quando mais seco, isto é, com casas e postos de combustível, maior probabilidade terá).

O Quartel possui uma lista dos fogos ativos que já estão confiados a algum bombeiro, e uma lista de fogos ativos que ainda estão à espera de serem delegados. Assim, a cada momento da simulação, o Quartel consulta a lista de fogos em espera e começa por atribuir bombeiros aos fogos com maior **gravidade** (que varia de acordo com a distância às habitações, e que estará constantemente a ser reavaliada). O Quartel delegará essa tarefa com base em:

1. Zona onde o incêndio se situa: o agente participativo deverá fazer parte dessa zona;
2. Distância do agente participativo ao incêndio (considerando a velocidade máxima do respetivo tipo de veículo);
3. Ter água suficiente para combater o determinado incêndio.

Depois de enviado o pedido ao respetivo bombeiro, o mesmo poderá aceitar ou recusar o encargo. Para tomar esta decisão, o próprio bombeiro terá em conta o seu estado interno. Desta forma, se estiver em repouso ou a regressar à sua posição ideal, o bombeiro irá **aceitar** a tarefa, caso contrário, se estiver já a dirigir-se para outro incêndio ou ocupado a combater algum, irá **recusar** a tarefa.

Quando um veículo está a dirigir-se ou a regressar de um incêndio e, por mero acaso, aproxima-se demasiado de um outro incêndio, o bombeiro pode **proativamente** optar por combater esse

fogo. Desta forma, este comunicará a sua intenção ao Quartel, que irá avisar o bombeiro que já estava atribuído ao fogo em questão para regressar para a sua posição ideal (caso exista).

Assim que um veículo acabe de combater um incêndio poderá ir **reabastecer** o seu depósito de água, caso esteja a metade ou menos de metade, abastecendo também o seu depósito de combustível se necessário e, por fim, regressar à sua posição ideal.

No entanto, tal como já referido, quando o Quartel envia um pedido a um agente participativo para combater um incêndio existe a possibilidade da tarefa ser rejeitada. Neste caso, o Quartel pedirá ao segundo veículo que melhor satisfaz as três condições anteriormente mencionadas para combater o fogo. A lista de bombeiros vai sendo percorrida **ciclicamente** até algum dos bombeiros aceitar a tarefa, e, entretanto, o fogo continuará na lista em espera.

Existe ainda um outro fator que terá que ser tido em conta pelo Quartel no processo de monitorização das catástrofes: a **taxa de ocupação** da zona. Esta taxa representa o número de fogos a dividir pelo número de agentes participativos presentes numa zona, que, consequentemente, pode ser interpretada como o excesso de incêndios em curso para a quantidade de recursos disponível. Neste caso, se a taxa for demasiado alta (maior que 150%), o Quartel realocará temporariamente recursos de zonas com uma menor taxa. Nesta alocação cada bombeiro terá somente como objetivo combater um determinado incêndio, e deverá regressar para a sua posição ideal no final.

4 Arquitetura

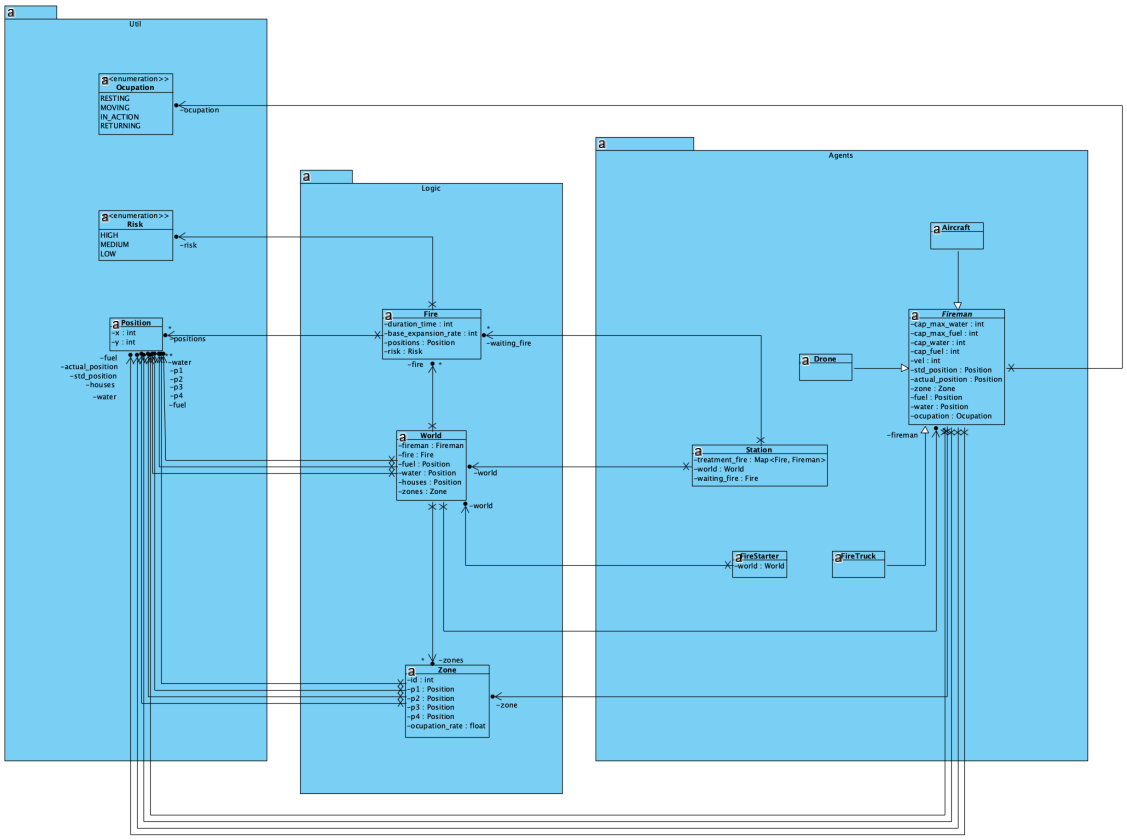


Figura 2: Diagrama de Classes.

Para a modelação do nosso sistema multiagente começamos por elaborar um diagrama de classes, presente na Figura 2. Em seguida, explicamos as diferentes classes e as respetivas variáveis.

O estado do simulador é da responsabilidade da classe ***World***. A disposição do mapa engloba a **lista dos bombeiros**, a **lista dos fogos** existentes, a **lista das posições** dos postos de abastecimento de **combustível**, **água** e de **habitações**. Por fim, possui ainda a **lista das zonas** existentes no mapa.

A ***Zone*** tem conhecimento relativo ao seu **id**, às coordenadas das **quatro posições** (vértices) que delimitam a respetiva zona e à sua **taxa de ocupação**, isto é, o número de fogos a dividir pelo número de agentes participativos presentes na zona (se a taxa for demasiado alta, terão que ser enviados temporariamente novos agentes participativos para a zona).

O ***Fire*** tem uma **lista de posições** (poderá ser mais do que uma quando o fogo se alastra), uma **gravidade** (alta, média ou baixa, uma vez que certas posições poderão apresentar habitações nas redondezas), o **tempo de duração** que o incêndio se apresenta ativo e, por fim, a **taxa de expansão base** do fogo (relacionada com o ambiente; por exemplo, se tiver próximo de uma zona com água, tem menor probabilidade). A taxa de expansão aplicada será baseada na taxa base e no tempo de duração do fogo.

O Agente Central, ***Station***, apresenta conhecimento relativo à **disposição do mapa**, nomeadamente as posições das habitações, dos locais de abastecimento de combustível e de água, as delimitações das zonas e as informações relativas aos fogos e aos bombeiros. Para além disso, armazena uma **lista de todos os fogos ativos**,o respetivo agente participativo encarregue pelo mesmo e uma **lista de todos os fogos que estão em espera** (sem nenhum agente encarregue por apagá-lo).

O Agente Secundário, ***FireStarter***, tem acesso ao **estado do jogo**, o que significa que tem acesso às posições em que não pode atear fogos (habitações, postos de abastecimento de água e combustível e posição dos bombeiros).

O Agente Participativo, ***Fireman***, tem como variáveis a sua **posição ideal** (aquela que lhe é atribuída no início da geração do mapa), a sua **posição atual**, a **zona** a que pertence, uma **lista das posições dos postos de abastecimento de água e combustível** da sua zona, a **capacidade máxima** e os **níveis atuais de combustível e água**, a **velocidade de movimento**, e, por fim, a sua **ocupação atual**, que poderá ser "Em repouso" (quando está na sua posição ideal e não lhe está atribuído nenhum fogo), "Em deslocação" (quando está a dirigir-se para o fogo), "Em ação" (quando está a apagar o fogo) e "A regressar" (quando está a regressar para a sua posição ideal).

O ***Fireman*** poderá ser um ***FireTruck***, um ***Drone*** ou um ***Aircraft***.

A ***Position***, é um par ordenado (X,Y) e corresponde a uma posição do mapa.

5 Protocolo de Comunicação

Como seria de esperar, no simulador os agentes irão comunicar entre si de modo a provocarem e resolverem catástrofes naturais.

Para representar estas interações recorremos a diagramas de sequência de sistemas, que refletem as diferentes mensagens que são trocadas entre os agentes, nos diversos momentos da simulação. Estas comunicações serão baseadas nas normas FIPA utilizadas pelo *JADE*, ou seja, utilizaremos *performatives* para indentificar os vários tipos de pedidos envolvidos entre os agentes.

5.1 Inicia Agente

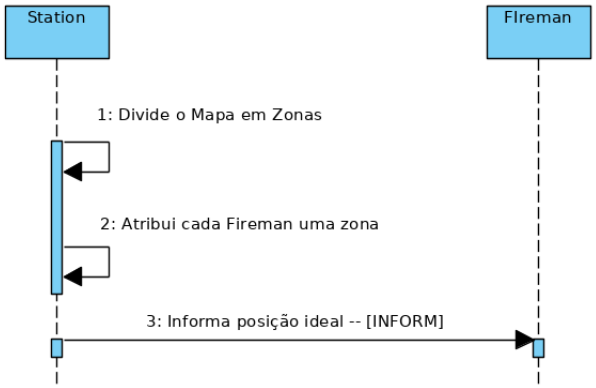


Figura 3: Diagrama de Sequência correspondente ao início de um bombeiro.

Ao iniciar a simulação, o quartel terá de começar por dividir o mapa em várias zonas e atribuir a cada uma destas um número de bombeiros, dos diferentes tipos. Após a seleção, indicará a cada um a sua posição ideal, ou seja, a posição onde deverá permanecer sempre que não estiver em combate.

5.2 Agente Informa Estado Atual

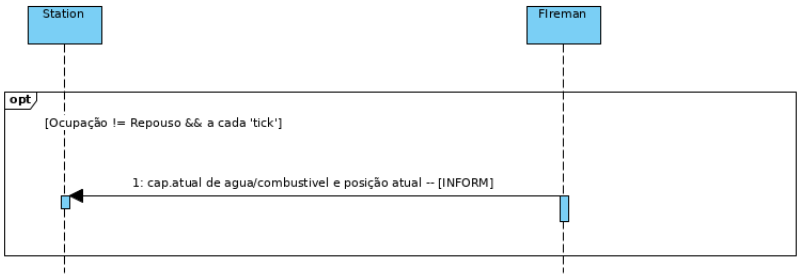


Figura 4: Diagrama de Sequência correspondente à comunicação do bombeiro sobre o seu estado ao quartel.

Durante a simulação o quartel terá de saber o estado de cada um dos bombeiros. Por isso, sempre que este estiver ocupado, seja em deslocação ou em combate - se estiver em repouso não terá nada para informar -, e a cada momento da simulação, este enviará uma mensagem a indicar os seus níveis de água e combustível, assim como a sua posição atual.

5.3 Ateador Inicia Fogo

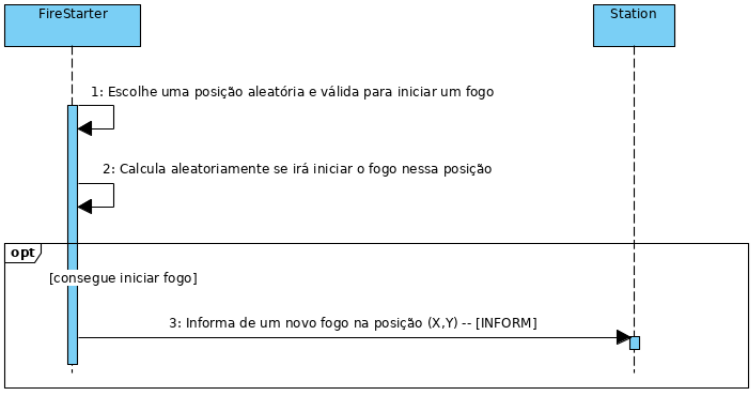


Figura 5: Diagrama de Sequência correspondente à criação de incêndios.

O Ateador de fogos, a cada momento da simulação tenta iniciar um novo fogo. Trata-se somente de uma tentativa, pois apesar de determinar uma posição, válida, para o novo fogo, apenas o concretiza se houver probabilidade (aleatória e variável) para tal.

5.4 Deteta Fogo

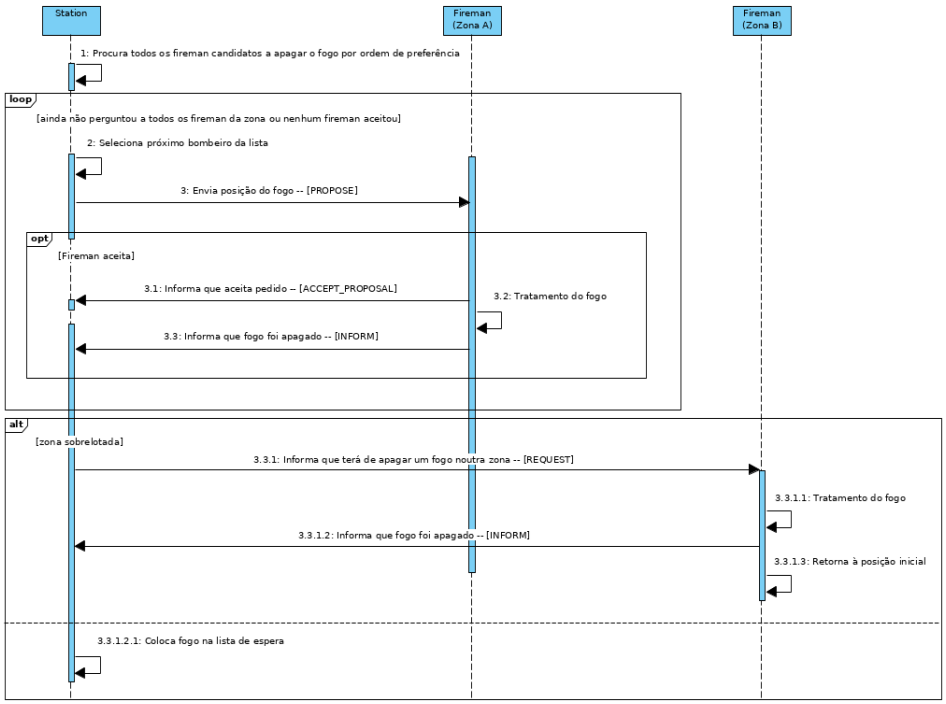


Figura 6: Diagrama de Sequência correspondente à deteção de um fogo.

A cada momento da simulação o Ateador de Fogos irá "tentar" iniciar um novo fogo. Assim que um novo aparecer, o quartel irá detetá-lo e proceder ao seu tratamento. Após determinar o bombeiro mais apropriado dentro da zona onde se encontra o fogo, este irá enviar-lhe uma mensagem a perguntar se aceita apagar o fogo. Se o bombeiro aceitar, notifica o quartel que aceitou, trata o fogo e informa de novo que concluiu a tarefa. Porém, caso não aceite, pergunta ao próximo candidato. Por fim, chegando ao fim da lista e nenhum ter aceite, este verifica se a zona está sobrelotada com fogos e, caso esteja acima de aproximadamente 150%, aloca um novo agente de uma outra zona não sobrelotada para temporariamente sair da sua zona e apagar o fogo em questão. Se não estiver sobrelotada, simplesmente é colocado em lista de espera para mais tarde ser reavaliado.

5.5 Iniciativa de Apagar um Fogo do Agente

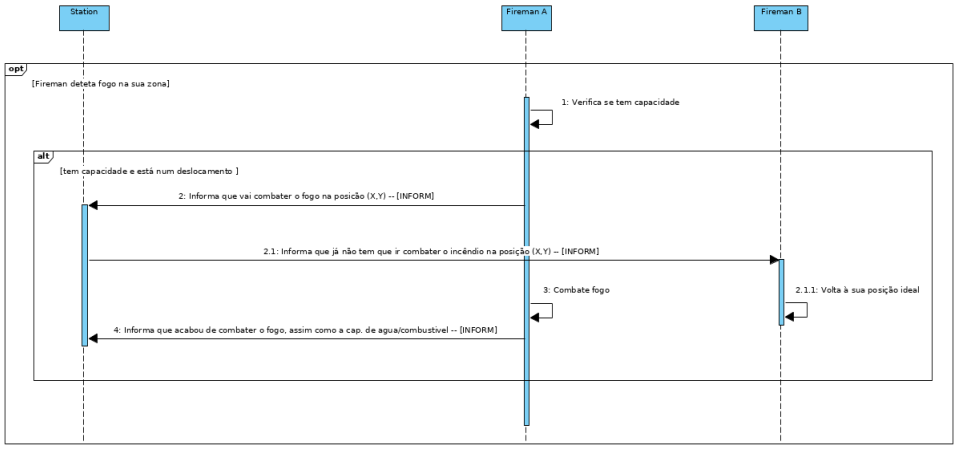


Figura 7: Diagrama de Sequência correspondente à resolução de um fogo por iniciativa do bombeiro.

Se um bombeiro estiver em regresso de outro fogo ou a caminho de algum, este poderá tomar a iniciativa de apagar um fogo que esteja perto, com algumas condições. Se resolver apagá-lo fá-lo-á depois de notificar o Quartel que irá apagar o respetivo fogo que, por sua vez, informará o bombeiro responsável por este fogo (se houver algum atribuído) que já está a ser tratado e cancelar a ação.

6 Processamento Interno dos Agentes

Os diagramas seguintes são demonstrativos do processamento interno de cada um dos agentes envolvidos no sistema.

6.1 Agente Incendiário

Como podemos ver no diagrama este agente só possui dois estados, em repouso e a *incendiar*. O Agente, após se iniciar o programa, fica em repouso até decidir preparar um incêndio. Após esta transição o agente fica no estado a *incendiar*. Quando o programa é terminado, o agente termina consequentemente.

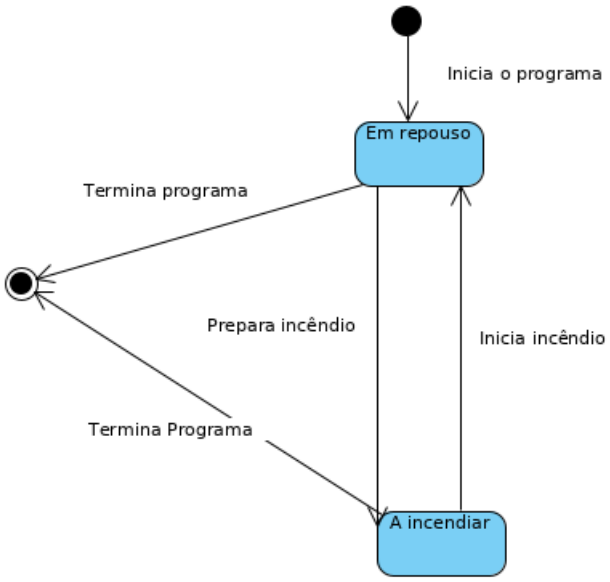


Figura 8: Máquina de estados do Agente Incendiário

6.2 Agente Quartel

O quartel após iniciar o programa entra num primeiro estado que apenas ocorre dessa vez, calcular posições ideais dos agentes Bombeiro. Após o término desse cálculo os estados bifurcam-se no cálculo de taxas de ocupação das zonas e de risco de incêndio, caso existam, e na procura de incêndios. Assim que é detectado um incêndio transita para o estado de procura do melhor agente para o combate, após encontrar este agente envia-lhe uma mensagem de pedido de ação. Depois disto fica em espera até receber uma resposta. Caso seja positiva transita para a bifurcação inicial, caso seja negativa retorna ao estado de procura de agente. Tal como no agente anterior, caso o programa seja terminado o agente termina com ele.

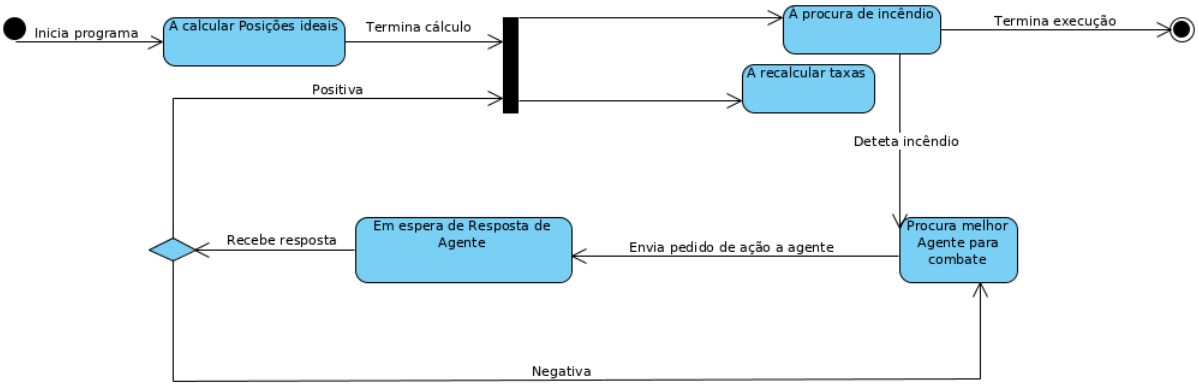


Figura 9: Máquina de estados do Agente Quartel

6.3 Agente Bombeiro

O agente bombeiro, tal como já foi referido nas secções anteriores, apresenta quatro estados. Após o programa ser inicializado ele recebe a sua posição ótima e começa o deslocamento para essa posição. A partir do momento que chega ao destino fica em espera até receber o pedido de ação, podendo aceitar ou recusar esse pedido. Caso aceite começa a deslocar-se para a posição do incêndio, caso recuse continua em espera. Ao deslocar-se para um incêndio, aquando da chegada passa para o estado em ação. Posteriormente a terminar o combate começa a regressar para a sua posição ideal. No regresso ou a caminho de um fogo, se detetar um incêndio pode, por iniciativa própria, combater esse fogo.

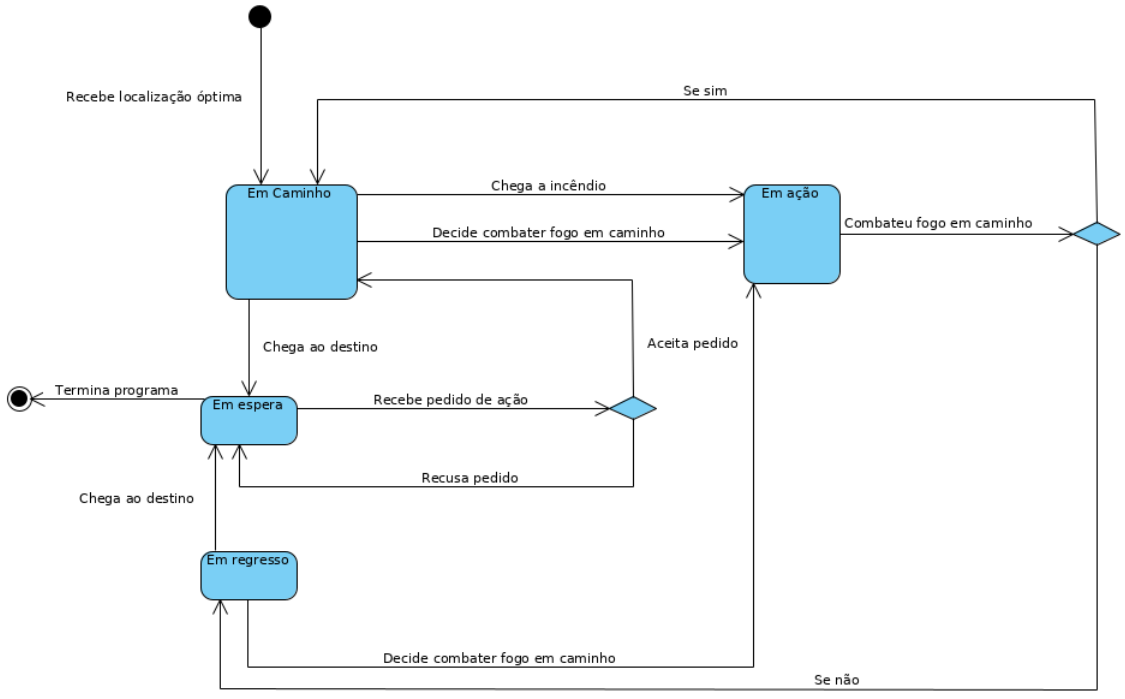


Figura 10: Máquina de estados do Agente Bombeiro

7 Mudanças em relação à 1.ª Fase

Uma das principais alterações em relação à 1ªFase , foi a necessidade de criação de uma estrutura de dados semelhante à do nosso agente *Fireman* para guardar os dados de forma que a *Station* tenha acesso ao estado dos agentes que combatem os fogos sempre que necessário.

Todo o resto da implementação foi baseada na modelação da fase anterior.

8 Implementação dos Agentes

Arquiteturalmente foram definidas uma série de diretrizes que os agentes têm de seguir:

- ***Behaviours***:
  - Tanto o agente *Fireman* como o *Station* têm um comportamento responsável pela receção e tratamento de mensagens, trata-se de um comportamento cíclico;
  - O agente *Fire Starter* tem um comportamento associado para informar a *Station* sempre que ateou um fogo;
  - O movimento de um agente do tipo bombeiro é definido através de um comportamento do tipo *Ticker*;
  - O envio das mensagens iniciais a cada um dos agentes relativamente aonde se posicionar também foi tratado com um comportamento único, executado no início da simulação, dado que é a *Station* que calcula todas essas posições;
  - A atualização da taxa de fogos por agentes numa zona é também um comportamento;
  - Existem outros comportamentos importantes para o funcionamento da aplicação, verificar os fogos em espera e o cálculo do risco associado aos fogos ativos ambos cíclicos. Estes comportamentos são executados pela *Station* pois é o único agente que tem acesso a todos os dados necessários.
- As mensagens enviadas entre os todos os agentes do sistema contêm **Objetos** para transportar a informação, sendo que estas também são únicas e distintas para as diferentes ações no sistema.

Além destes comportamentos principais existem também outros, aos quais chamamos de *handlers*, visto que o seu propósito é após a descodificação da mensagem executar o que lhe é pedido, ou têm programado para fazer.

Entre estes destacam-se:

- Começo de um incêndio;
- Aceitar ou recusar pedido de combate a um fogo;
- O envio da informação inicial ao agente bombeiro.

Outro facto importante acerca destes comportamentos é que são todos do tipo *OneShot*. Isto deve-se ao facto de serem respostas ao que foi inferido através do processamento das mensagens.

8.1 *Station*

O *Station* é o agente que tem conhecimento acerca de tudo o que se passa no mundo. É ele que atribui as tarefas a cada um dos agentes *Fireman* para a resolução eficaz e gastar o mínimo de recursos possíveis. Dentro do agente *Station* existe um método fulcral: o de decisão do bombeiro mais adequado para resolver um determinado fogo. Neste, após descobrir a zona em que o fogo se encontra, procura todos os bombeiros associados a esta zona, ordena-os por distância ao fogo e por fim, por tamanho dos tanques de água comparado com o tamanho do fogo. No fim, envia a mensagem a questionar o bombeiro se aceita ou não, sendo que caso a resposta seja negativa, este

volta a correr o algoritmo retirando primeiro o bombeiro questionado anteriormente da lista de possíveis candidatos. Este também é responsável por atualizar os estados dos fogos (expansões, tempo ativos, risco associado) assim como notificar todos os agentes de mudanças nestes. No caso do risco de um fogo, ele verifica se existem edifícios à volta deste e, dependendo da sua distância ao fogo, terá 3 tipos de risco: *LOW, MEDIUM, HIGH* e, no caso das expansões, para cada fogo no mundo, é calculado a probabilidade de poder expandir naquele instante (sendo que cada fogo tem uma taxa de início de expansão associada, conforme o ambiente em que se encontra) e, se alta, expande para uma posição disponível à sua volta. Para além disto, mas talvez não tão crucial para o funcionamento dos agentes, o *Station* é responsável também por gerar a interface com o mapa (utilizando as informações que já contém acerca do mundo), informar do estado do mapa aos *Fireman* e por gerar a interface com algumas estatísticas sobre a simulação.

Na figura seguinte podemos verificar algumas *Performatives* que este agente consegue processar.

```
public void action() {
        Station s = (Station) myAgent;
        ACLMessage msg = myAgent.receive();
        if (msg == null) {
            block();
            return;
        }
        try {
            Object content = msg.getContentObject();
            switch (msg.getPerformative()) {
                case (ACLMessage.INFORM):
                    if (content instanceof UpdateData)
                        s.addBehaviour(new HandleUpdateData(s, msg));
                    else if (content instanceof FireExtinguished) {
                        s.addBehaviour(new HandleFireExtinguished(s, msg));
                    } else if (content instanceof StartedFire) {
                        s.addBehaviour(new HandleFireStarted(s, msg));
                    }
                    break;
                case (ACLMessage.AGREE):
                    if (content instanceof ExtinguishFireData)
                        s.addBehaviour(new HandleAcceptRequest(s,msg));
                    break;
                case (ACLMessage.REFUSE):
                    if (content instanceof ExtinguishFireData)
                        s.addBehaviour(new HandleRefuseFireRequest(s,msg));
                    break;
                default:
                    System.out.println("Wrong message content.");
                    break;
            }
        } catch (UnreadableException e) {
            System.out.println("Error Station on msg: " + msg.getPerformative() + " -> " +
                msg.getContent());
        }
    }
```

8.2 *Fireman*

Sendo que o objetivo de um *fireman* é apagar um fogo, este tem de se saber movimentar pelo mapa. Por isso, para além do conhecimento que tem de possuir sobre o estado do mapa a cada instante, tem a pesada tarefa de encontrar o caminho mais curto para um fogo, certificando-se que possui combustível que chegue para chegar a este e voltar, assim como reabastecer água e combustível sempre que necessário.O algoritmo utilizado para determinar os caminhos mais curtos é o *breadth first search*.

Desta forma, quando lhe é atribuído um fogo, o bombeiro começa por testar se tem combustível suficiente para ir combater o fogo e abastecer no final no posto de combustível mais próximo do fogo. Em caso negativo, o bombeiro averigua se tem combustível suficiente para primeiramente abastecer no posto mais próximo do fogo, e, em seguida, apagar o fogo. Em caso negativo, o bombeiro opta por abastecer no posto de combustível mais próximo deste.

Assim que apaga um fogo, o bombeiro regressa para a sua posição standard, mas se estiver com o depósito de água a metade ou menos de metade começa por se dirigir ao posto de abastecimento de água mais próximo e só depois é que retorna para a sua posição standard. Neste objetivo intermédio, o bombeiro faz constantemente averiguações de combustível e dirige-se ao posto de combustível mais próximo deste ou do local de abastecimento de água, caso seja necessário.

Também existente neste agente, encontra-se a possibilidade de este ser pró-ativo e apagar um fogo se estiver a caminho e/ou a regressar de outro. Para saber da existência de fogos no mapa, o *Station*, a cada novo fogo ou expansão deste, notifica todos os agentes do mapa com a nova informação. Depois, à medida que executa um movimento, se detetar um fogo num raio de 2 casa em qualquer direção e possuindo água para o apagar, ele procede em apagá lo e a notificar o *Station* do que fez.

Na figura seguinte podemos verificar algumas *Performatives* que este agente consegue processar.

```
public void action() {
       Fireman f = (Fireman) myAgent;
       ACLMessage msg = myAgent.receive();
       if(msg == null){
           block();
           return;
       }
       try{
           Object content = msg.getContentObject();
           switch (msg.getPerformative()){
               case(ACLMessage.INFORM):
                   if(content instanceof InitialData) {
                       f.addBehaviour(new HandleInitialData(f, msg));
                   }
                   else if(content instanceof UpdateFire){
                       f.addBehaviour(new HandleUpdateFire(f,msg));
                   }

                   break;
               case(ACLMessage.REQUEST):
                   if(content instanceof ExtinguishFireData){
                       f.addBehaviour(new HandleExtinguishFireData(f,msg));
                   }
                   break;
               case(ACLMessage.CANCEL):
                   if(content instanceof CancelFire)
                       f.addBehaviour(new HandleCancelFire(f,msg));
                   break;
               default:
```

```
                System.out.println("Wrong message content.");
                break;
            }
        } catch (UnreadableException e) {
            System.out.println("Error Fireman on msg: " + msg.getPerformative() + " -> " +
                msg.getContent());
        }
    }
```

8.3 *Fire Starter*

Existe no sistema um agente que o seu único objetivo é criar fogos no mapa e avisar a *Station* de que criou um fogo. Este agente gera uma posição aleatória, verifica se ela é válida e, só depois informa o outro agente.

9 Interface

9.1 Mapa

O mapa apresenta as diferentes zonas em que o mundo se encontra dividido, cada um com a sua cor, e imagens de todos os tipos de entidades no mundo, tal como se pode ver na Figura 13. Existe também uma versão no código onde os recursos se encontram apresentados em diferentes cores, nomeadamente:

- Edifícios: cor branca;
- Água: cor azul;
- Combustível: cor cinzento;
- Fogos: cor vermelha;r
- Drones: cor amarela;
- Aviões: cor ciano;
- Camiões: cor lilás;
- Os Bombeiros possuem um número associado ao seu nome no sistema, para melhor visualização.

Esta segunda versão, presente na Figura 12, só existe para facilitar o *debug*.

Como foi referido anteriormente o mapa é atualizado através de um comportamento *Ticker* no agente *station* porque só desta forma é possível assegurar que as alterações mostradas no mapa estão consistentes e corretas.

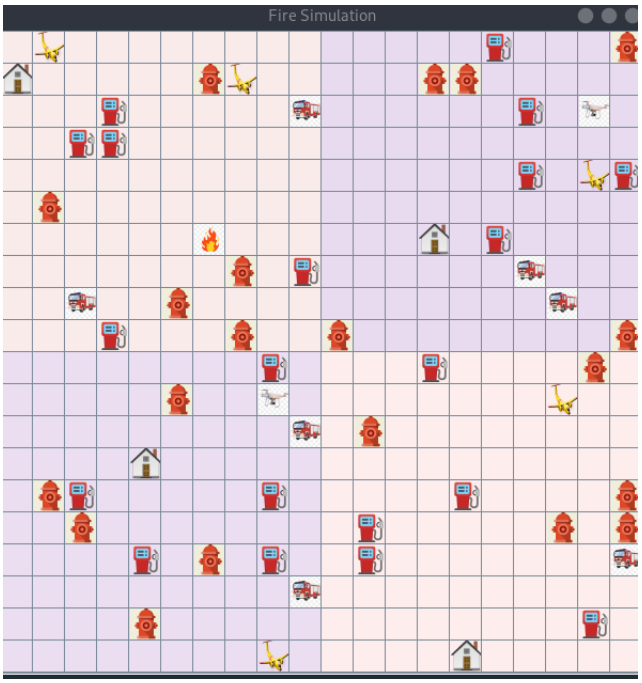


Figura 11: Mapa

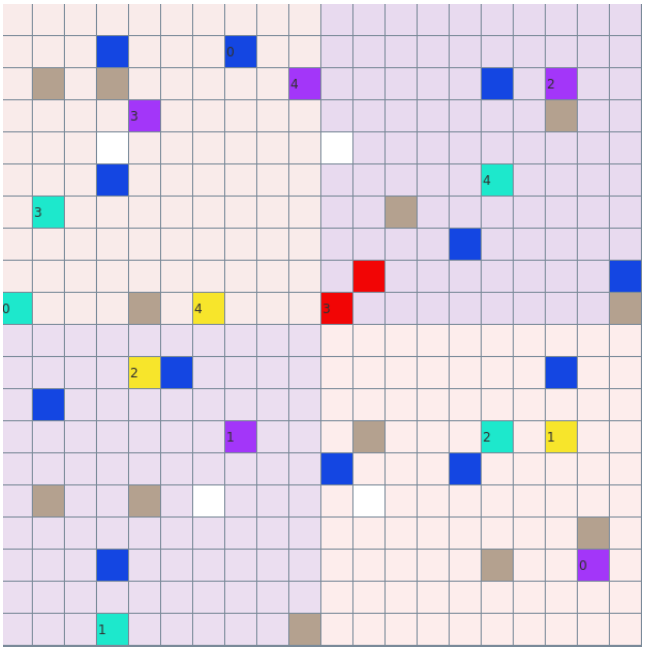
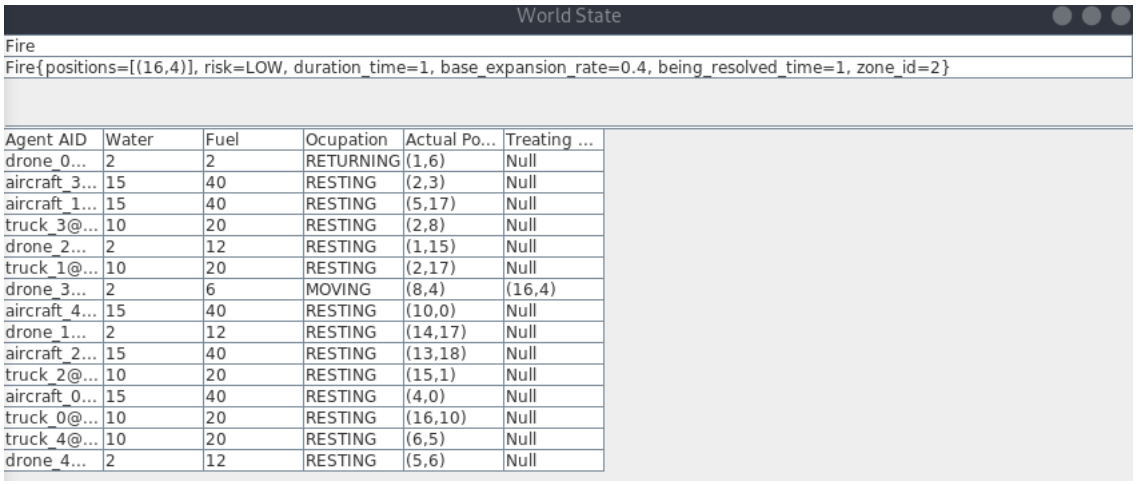


Figura 12: Mapa versão debug

9.2 Estado dos agentes e dos fogos

Para além do mapa, é mostrado também, do ponto de vista do *Station*, o estado de todos os agentes no sistema, desde a sua ocupação, capacidade de água e combustível, posição atual e fogo que estão a combater. Também é possível visualizar o estado dos fogos.



Agent AID	Water	Fuel	Ocupation	Actual Po...	Treating ...
drone_0...	2	2	RETURNING	(1,6)	Null
aircraft_3...	15	40	RESTING	(2,3)	Null
aircraft_1...	15	40	RESTING	(5,17)	Null
truck_3@...	10	20	RESTING	(2,8)	Null
drone_2...	2	12	RESTING	(1,15)	Null
truck_1@...	10	20	RESTING	(2,17)	Null
drone_3...	2	6	MOVING	(8,4)	(16,4)
aircraft_4...	15	40	RESTING	(10,0)	Null
drone_1...	2	12	RESTING	(14,17)	Null
aircraft_2...	15	40	RESTING	(13,18)	Null
truck_2@...	10	20	RESTING	(15,1)	Null
aircraft_0...	15	40	RESTING	(4,0)	Null
truck_0@...	10	20	RESTING	(16,10)	Null
truck_4@...	10	20	RESTING	(6,5)	Null
drone_4...	2	12	RESTING	(5,6)	Null

Figura 13: Estado dos agentes bombeiros e dos fogos

9.3 Estatísticas

Em termos de estatística, o programa gera uma janela com 4 gráficos distintos. O primeiro trata de mostrar, ao longo do tempo, o número de fogos em tratamento e o número de fogos em espera, útil para ter uma noção do estado do mapa, se há sobrecarga nos bombeiros, etc. O segundo apresenta, ao longo do tempo, o número de segundos médio para atribuição de um novo fogo a um bombeiro e também do tratamento deste, indicando nos o quão eficaz estão realmente a ser os agentes no mundo. Por fim, os últimos dois são meramente dados indicativos: número de fogos por zona a cada instante e tipos de veículo em uso.

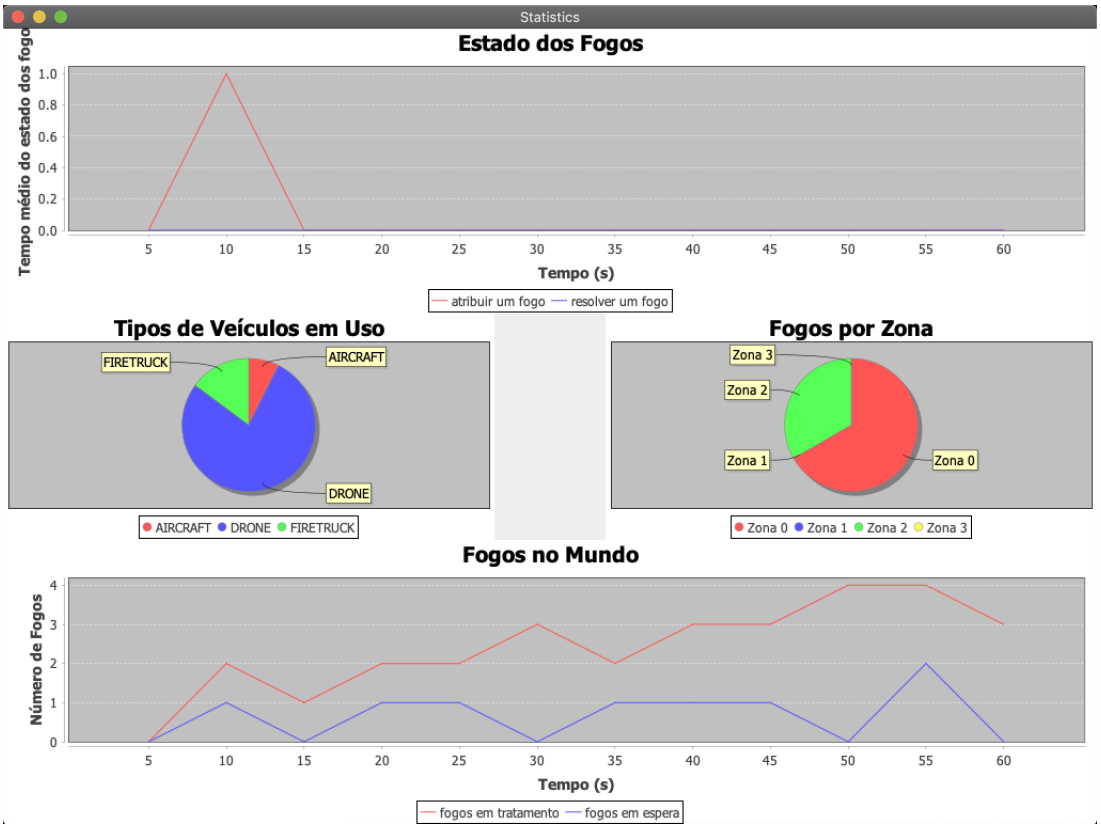


Figura 14: Estado dos agentes bombeiros e dos fogos

10 Caso de Estudo

Para testar o simulador, foi gerado um mapa 40x40, com 8 tipos de cada bombeiro, espalhados por 4 zonas, com 8 postos de combustível, de água e edifícios. Infelizmente não nos é possível gerar mapas de maior dimensão, pois as nossas máquinas não são capazes de aguentar o processamento, bloqueando assim a execução.

Com uma geração de aproximadamente 1 fogo a cada 2 segundos, mais expansões de cada fogo a decorrerem a cada 1 segundo, verificamos que o tipo de veículo mais utilizado é o *Drone*, sendo que a resolução dos fogos são rápidas e eficazes (devido à sua velocidade de deslocação). Infelizmente, ao longo da execução, pode acontecer os bombeiros ficarem sem opções de deslocação e ocasionalmente ficarem sem combustível aquando do combate de um fogo. Isto provoca um efeito de alastramento do fogo, sem poder controlar. Se neste exemplo aumentarmos para 12 postos de combustível, notamos numa melhoria de performance. Importante referir que também depende muito do momento de execução, porque pode acontecer os postos serem gerados em posições pouco estratégicas.

Como referido antes, as resoluções dos fogos são feitas relativamente depressa (em média demoram 1 segundo a resolver) logo, os fogos são apagados ainda jovens (com apenas 1 posição).

11 Conclusão e Trabalho Futuro

O planeamento do sistema multiagente que visa monitorizar e resolver catástrofes naturais foi, sem dúvida, uma fase determinante para o futuro processo da implementação do simulador.

A tarefa mais complicada ao longo da idealização do sistema foi a tentativa de equilibrar as responsabilidades entre o Quartel (Agente Central) e os Veículos Bombeiros (Agentes Participativos). Desta forma, o objetivo passou por tentar não sobrecarregar o Quartel na seleção do bombeiro mais adequado para resolver o incêndio, visto que existe um vasto leque de variáveis que podem ser consideradas nesta decisão, tais como: a capacidade atual de combustível e de água do bombeiro, a distância do mesmo em relação ao incêndio considerando também as distâncias caso ele tenha que abastecer ao longo caminho (o que faria com que o Quartel tivesse que projetar à partida o trajeto a ser efetuado pelo bombeiro), ou ainda considerar distâncias cruzadas, de forma a que fosse encontrada a solução ótima global.

Como não desejamos centralizar demasiado a resolução dos incêndios, mas mesmo assim não querendo prejudicar excessivamente a eficiência da solução, tentamos que o agente participativo interferisse inteligente e ativamente nas decisões. Como tal, optamos por considerar que o Quartel apenas teria em conta a distância em relação ao local do incêndio e se o veículo tem água suficiente para o apagar, desprezando os restantes fatores. O bombeiro tem a possibilidade de recusar o pedido e de proativamente combater um incêndio, caso se cruze com um numa deslocação. Este modelo de deteção e resolução de catástrofes certamente que não conduz o nosso simulador à solução ótima, mas torna os nossos agentes mais autónomos e cooperativos.

Outra das decisões tomadas foram a divisão do mapa por zonas, o que faz com que a escolha do melhor veículo seja mais limitada. Porém, consideramos essa limitação benéfica, uma vez que acreditamos que tornará o mapa mais organizado e civilizado.

No que toca à implementação de todas as funcionalidades descritas no relatório, podemos concluir que todas foram realizadas com sucesso. Existiram dificuldades no desenvolvimento pois manter a persistência e a coerência dos dados no momento das comunicações entre os vários agentes torna-se uma tarefa muito complexa e demorada. Ainda mais complexa é esta gestão quando os agentes que são incubidos de tarefas, neste caso os *Fireman*, por outro agente numa hierarquia superior, a *Station*, podem recusar e tomar ações próprias.

Uma dificuldade com que nos deparamos foi a correção de erros e comportamentos indefinidos no desenvolvimento do projeto. Como em qualquer sistema, ao ser desenvolvido vão ser encontrados erros. Consideramos que a plataforma JADE não proporciona o melhor ambiente de *debug*, não sendo este intuitivo nem fácil de usar.

No caso da nossa implementação, devido ao algoritmo de movimento de bombeiro, *breadth first search*, ser bastante pesado, faz com que a dimensão do mapa não possa ser superior a aproximadamente 100, pois o CPU das nossas máquinas não aguenta.

Como trabalho futuro o objetivo é tornar o simulador o mais verídico possível, adicionando, por exemplo, estradas ao mapa, que os fire trucks seriam obrigados a utilizar. Os bombeiros também poderiam ter a possibilidade de armazenar combustível para fornecerem a outros bombeiros. Um pormenor importante que poderia ser adicionado é, caso um bombeiro tenha o azar de ficar sem combustível no combate a um fogo, notificaria o quartel para enviar outra pessoa.

Referências

[1] Muhammad Aslam, Muhammad Tariq Pervez, Syed Shah Muhammad, Seemal Mushtaq, Ana Maria Martinez Enriquez, *FMSIND: A Framework of Multi-Agent Systems Interaction during Natural Disaster*

[2] Xiangyu Si, Jonathan Li, Zhijun Wang, *Knowledge-Oriented Sensor Web For Disaster Management: From Sensing To Decision Making*

[3] F.Fiedrch, *An HLA-based multiagent system for optimized resource allocation after strong earthquakes*