Universidade do Minho
Escola de Engenharia
Mestrado em Engenharia Informática

Agentes e Sistemas Multiagente

Ano Letivo de 2022/2023

Gestão de Aeroporto

Daniel Xavier PG50310
Diogo Rebelo PG50327

20 de junho de 2023

ASM

Resumo

Como mote, este documento apresenta e descreve a fase de implementação da gestão de um aeroporto na cidade de Braga.

Por conseguinte, o objetivo deste trabalho prático foi desenvolver um sistema multiagente para a gestão de partidas e chegadas em um aeroporto. O sistema utiliza a biblioteca SPADE para a implementação dos agentes.

De modo sucinto, a Torre de Controlo é responsável pela gestão dos aviões que desejam aterrar ou descolar no aeroporto. Quando um avião deseja aterrar, ele informa a Torre de Controlo, fornecendo informações identificadoras do avião. A Torre de Controlo por sua vez consulta o Gestor de Gares do aeroporto para verificar se há uma área de estacionamento disponível (gare) para o avião. Além disso, deve verificar se há uma pista livre para a manobra de aterragem.

O Gestor de Gares é assim responsável por informar a Torre de Controlo sobre a disponibilidade de áreas de estacionamento para aviões.

O sistema de gestão do aeroporto também inclui um agente responsável por apresentar informações gerais sobre os voos em operação. Este agente reúne e disponibiliza informações sobre os aviões que estão para descolar ou aterrar e mantém contacto constante com a Torre de Controlo para manter as informações atualizadas.

A nossa aplicação encontrou nesta especificação uma base sólida para a sua robustez, desempenho e equilíbrio e sua implementação foi realizada seguindo esses mesmo pilares construindo uma arquitetura geral bem fundamentada.

Como tal, a nossa aplicação cumpre assim várias funcionalidades e o objetivo principal da gestão geral do aeroporto, respondendo assim às necessidades dos aviões.

Área de Aplicação: Engenharia de Software.

Índice

Lista de Figuras

1 Introdução

Tal como referido presente trabalho tem como objetivo explorar o tema de Agentes e Sistemas Multiagente aplicados à gestão de aeroportos. O desenvolvimento de sistemas multiagentes tem ganhado destaque na área de Engenharia de Software, devido à capacidade de modelar e simular interações complexas entre agentes autônomos em ambientes dinâmicos pelo que torna-se primordial o seu domínio.

Neste contexto, o projeto proposto consiste na conceção e implementação de um sistema multiagente para a gestão de partidas e chegadas em um aeroporto. A utilização da biblioteca SPADE para o desenvolvimento dos agentes proporciona uma base sólida para a criação de um sistema distribuído e inteligente.

O sistema multiagente a ser desenvolvido conta com a participação de quatro principais agentes: o Avião, a Torre de Controlo, o Gestor de Gares e o Agente de Informação de Voos. Cada agente desempenha um papel específico na coordenação das atividades aeroportuárias, desde o pedido de aterragem ou descolagem de aviões até a alocação de áreas de estacionamento e pistas.

A arquitetura do sistema procura assim otimizar a eficiência e a segurança das operações, considerando aspetos como a disponibilidade de recursos, a distância entre as gares e as pistas, e a comunicação constante entre os agentes para manter as informações atualizadas.

No decorrer deste relatório, serão abordados os detalhes da arquitetura proposta, as funcionalidades implementadas, os resultados obtidos, bem como a conclusão.

2 Arquitetura do Sistema

2.1 Diagrama de classes

É primordial numa primeira fase apresentar a arquitetura distribuída definida para este sistema. Esta, consiste em diferentes agentes que interagem de forma coordenada para realizar as tarefas necessárias. A estrutura do sistema é composta pelos seguintes componentes:

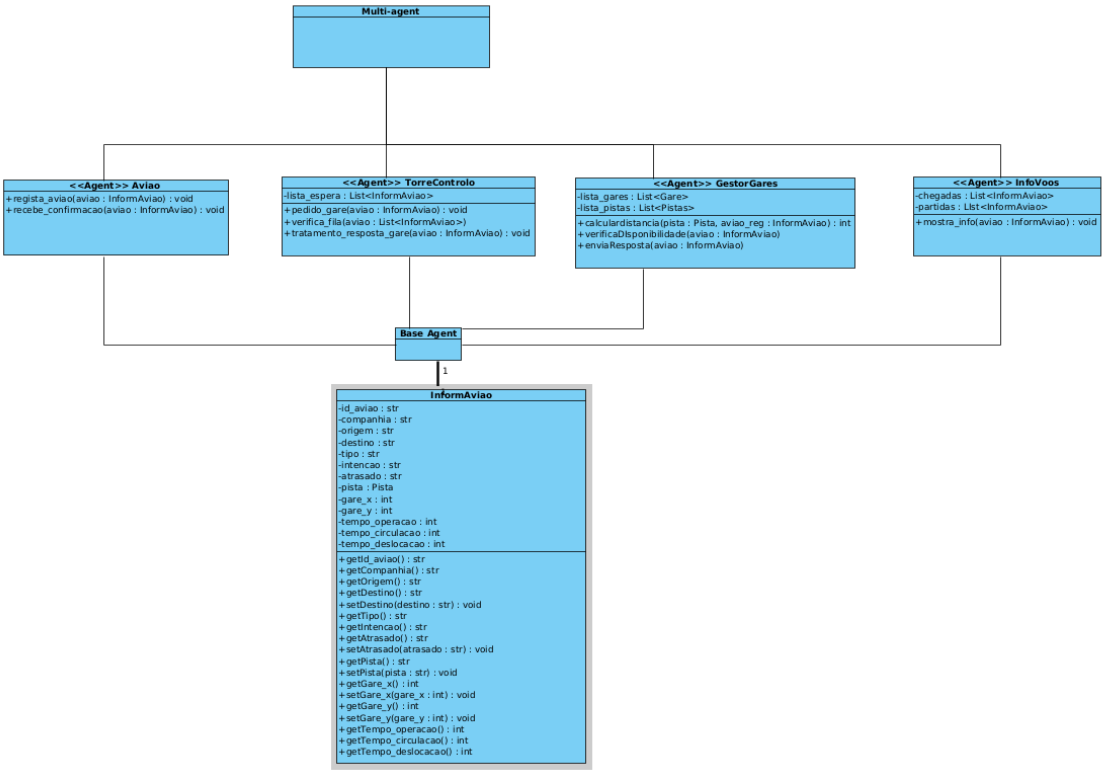


Figura 2.1: Diagrama de Classes

Deste modo serão de seguida analisados com mais detalhe os agentes e os seus comportamentos.

2.2 Agentes e Behaviours

2.2.1 Aviao

O agente avião é o principal agente do sistema uma vez que é o "óleo"para o motor e vai assim desencadear todos os comportamentos.

Ao ser inicializado, o agente configura e adiciona dois comportamentos:

- **RegistarAviao_Behav**: regista cada avião com as informações.
- **RecebeConfirmacao_Behav** recebe a confirmação para realizar a operação pretendida.

O comportamento RegistarAviao_Behav é um comportamento do tipo OneShotBehaviour, ou seja, é executado apenas uma vez. É responsável por gerar informações aleatórias do o avião, como companhia aérea, tipo (comercial ou carga), intenção (aterrar ou descolar), origem, destino, entre outros. Essas informações são encapsuladas num objeto "InformAviao"e enviadas para o agente "Torre de Controlo"através de uma mensagem com a *perfomative* request.

Por sua vez, o comportamento RecebeConfirmacao_Behav é um comportamento do tipo CyclicBehaviour, ou seja, é executado ciclicamente. Ele aguarda a receção de uma mensagem, com um timeout de 20 segundos. Quando a mensagem é recebida, o comportamento verifica o tipo de *performative* (confirm, refuse ou outra) para determinar a ação a ser tomada. Se a mensagem for de confirmação, o comportamento imprime informações sobre o avião e simula a operação do avião. Em seguida, envia uma mensagem para o agente Info Voos para informar sobre a conclusão do avião e outra mensagem para o agente Gestor de Gares para libertar a gare ocupada pelo avião. Se a mensagem for de recusa, o comportamento envia uma mensagem para o agente Info Voos a informar que o avião irá para outro aeroporto. Por outro lado,se nenhum mensagem for recebida dentro do tempo limite, o comportamento assume que o avião irá estacionar em outro aeroporto.

2.2.2 Torre de Controlo

Por sua vez, podemos dizer que a torre de controlo é o motor do sistema. Este é o principal agente coordenador das operações. Possuí dois comportamentos distintos:

- **RecebePedidoTorre_Behav**: coordena os pedidos dos aviões.
- **VerificaFilaEspera_Behav**: trata das filas de espera.

Relativamente ao primeiro e principal comportamento este comportamento é cíclico, ou seja, é executado continuamente enquanto o agente estiver ativo.

O comportamento aguarda assim receção de mensagens dos aviões e quando uma mensagem é recebida, o comportamento verifica a intenção do avião (aterrar ou descolar).

Após isso o *behaviour* envia uma mensagem para o gestor de gares para verificar se pode realizar a intenção do avião.

O comportamento também lida com as mensagens de confirmação ou recusa enviadas pelo gestor de gares. Se a confirmação for recebida, o comportamento envia uma mensagem de confirmação ao avião e notifica o gestor de gares para libertar a pista ou gare.

Se a recusa for recebida, o comportamento notifica o gestor de gares para libertar a pista ou gare e adiciona o avião à lista de espera, caso ainda não esteja presente.

Por outro lado também lida com a remoção de aviões da lista de espera quando recebem uma confirmação da torre de controlo.

Relativamente ao *behaviour* VerificaFilaEspera_Behav este comportamento é periódico, de 2 em 2 segundos.

O comportamento verifica se a lista de espera de aviões contém elementos.

Se a lista de espera não estiver vazia, o comportamento envia uma mensagem para o gestor de gares para verificar a disponibilidade de uma pista e/ou gare.

O comportamento lida com a resposta do gestor de gares e executa as ações apropriadas com base na disponibilidade.

2.2.3 Gestor de Gares

O gestor de gares implementa a lógica para gestão da disponibilidade de pistas e gares para aterragem e descolagem de aviões.

Possuí o comportamento:

- **EnviaRespostaGare_Behav**: gere as disponibilidades

O *behaviour* é executado ciclicamente. A mensagem recebida contém informações sobre a intenção do avião (aterrar ou descolar).

Se a intenção do avião for aterrar, o agente verifica a disponibilidade de pistas e gares. Primeiro, verifica se há uma pista disponível. Se houver, verifica a gare mais próxima e disponível para aterrar. Em seguida, envia uma mensagem de resposta indicando se o pedido foi aceite ou recusado.

Se a intenção do avião for descolar, o agente verifica a disponibilidade de pistas. Calcula a distância entre cada pista e a gare do avião e seleciona a pista mais próxima e disponível. Em

seguida, do mesmo modo envia uma mensagem de resposta indicando se o pedido foi aceite ou recusado.

O agente também trata mensagens de *release* de gares e pistas, atualizando o estado da ocupação das gares e pistas.

Além disso, o agente mantém uma lista de gares e uma lista de pistas disponíveis, bem como flags evitar conflitos no processo de aterragem ou descolagem.

2.2.4 Info Voos

Por último este agente é apenas responsável por manter a informação atualizada dos movimentos de todos os aviões no aeroporto. Apenas possuí um behaviour:

- **MostraInfoVoos_Behav**: exibe a informação dos voos

Este comportamento é cíclico, sendo executado continuamente enquanto o agente estiver ativo. Aguarda a receção de mensagens e quando uma mensagem é recebida, o comportamento verifica se o tipo de performativa é "inform"e posteriomente descodifica a mensagem num objeto da classe "InformAviao". O comportamento verifica a intenção do avião (aterrar ou descolar) e adiciona o avião à lista correspondente (chegadas ou partidas). Em seguida, o comportamento imprime uma tabela com as informações dos voos de chegadas e partidas, incluindo o ID do avião, companhia, origem/destino, tipo de voo, tempo de operação, tempo de circulação e tempo de deslocação. O comportamento também lida com a verificação de atrasos nos voos e imprime uma mensagem específica para voos atrasados ou mudanças de aeroportos.

2.3 Funcionalidades

De modo a melhorar identificação do trabalho realizado segue-se a lista das funcionalidades incorporadas:

- Definir um limite de aviões em fila de espera para aterrar no aeroporto. No caso de um avião querer aterrar, se o limite máximo estiver esgotado, será informado que terá de se dirigir para outro aeroporto. O limite máximo definido foi de 5 aviões;

- Sabendo que antes da descolagem, os aviões devem estar estacionados numa gare, ajustar o nº possível de aviões em espera, para estacionarem, ao número máximo de gares disponíveis;

- A Torre de Controlo é responsável por indicar a um avião que pretenda aterrar, qual a pista e a gare onde o deve fazer. Para tal, deverá ter em conta a distância entre a pista e as diferentes gares disponíveis, indicando a que tenha o caminho mais curto. No caso dos aviões que pretendem descolar, indicar ao Gestor de Gares qual a pista que pode utilizar, tendo em conta a distância mínima entre a gare e as pistas disponíveis;

- Definir um tempo de operação entre o avião e a pista de aterragem, um tempo de circulação na pista e um tempo de deslocação entre a pista e a gare. Foram assumidos valores aleatórios contudo que permitem realizar a simulação do sistema pretendido;

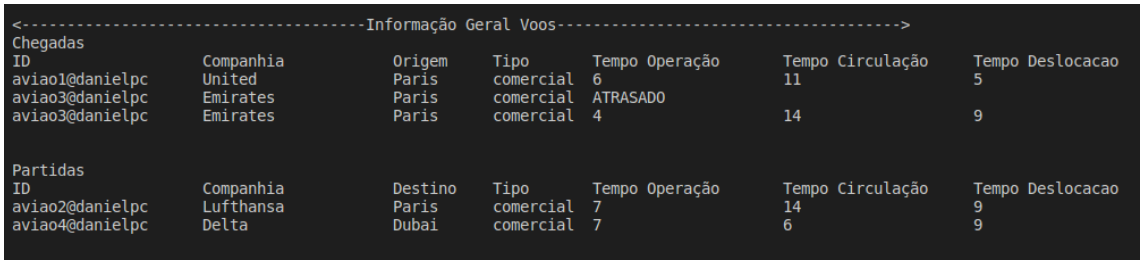
- Caso um avião que pretenda aterrar não obtenha resposta positiva para a manobra de aterragem por parte da Torre de Controlo ao fim de determinado tempo (40 segundos), deverá informar a mesma que irá efetuar a aterragem noutro aeroporto;

- Se ao mesmo tempo estiver uma gare disponível, e existir um potencial conflito entre um avião que pretenda aterrar e outro descolar, implementar mecanismos de prioridade para evitar estes conflitos. O mecanismo de prioridade foi definido como sendo o primeiro a fazer o pedido contudo há um controlo e o outro avião entrará na lista de espera.

  Foram assim incorporadas todas as funcionalidades obrigatórias contudo devido fundamentalmente à falta de tempo não foram implementadas as funcionalidades adicionais pelo que poderão ser um primeiro passo para o trabalho futuro.

3 Testes e Resultados

Numa fase inicial tal como proposto assumimos a existência de 2 aviões com intenção de aterrar e 2 em processo de descolagem. Posteriormente, escalamos estes números para valores mais realistas.



<----------Informação Geral Voos---------->

Chegadas

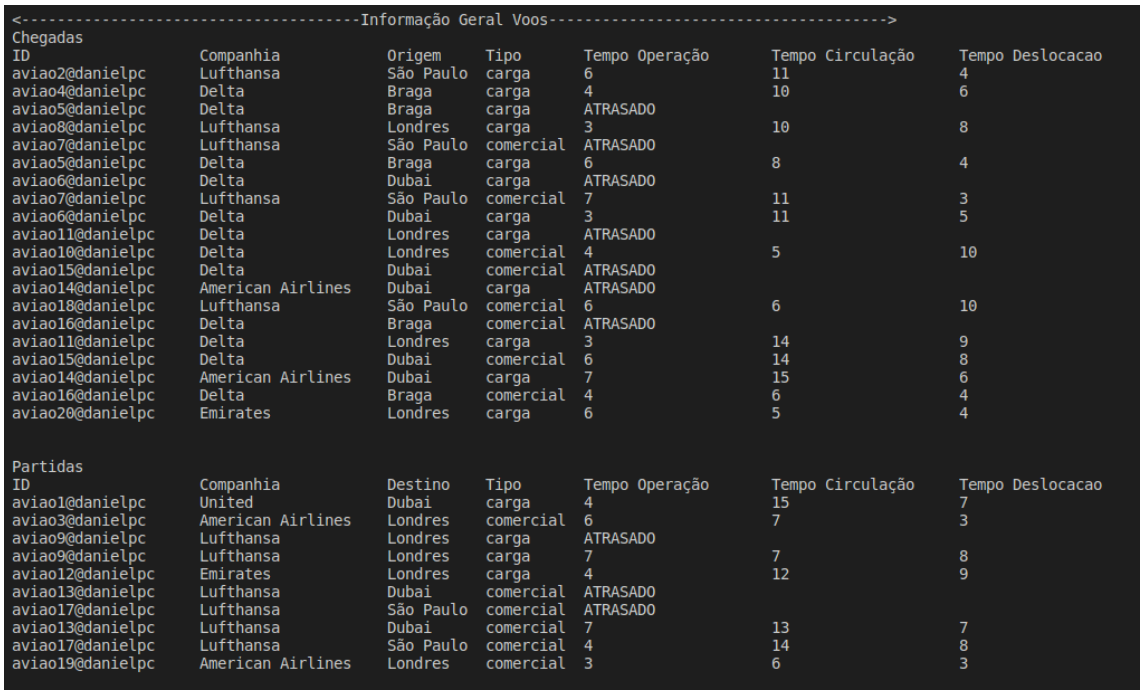
ID	Companhia	Origem	Tipo	Tempo Operação	Tempo Circulação	Tempo Deslocacao
aviao1@danielpc	United	Paris	comercial	6	11	5
aviao3@danielpc	Emirates	Paris	comercial	ATRASADO		
aviao3@danielpc	Emirates	Paris	comercial	4	14	9

Partidas

ID	Companhia	Destino	Tipo	Tempo Operação	Tempo Circulação	Tempo Deslocacao
aviao2@danielpc	Lufthansa	Paris	comercial	7	14	9
aviao4@danielpc	Delta	Dubai	comercial	7	6	9

Figura 3.1: Teste com 4 aviões

Com 20 aviões e com apenas 4 gares e 3 pistas o resultado final obtido foi o seguinte:



<----------Informação Geral Voos---------->

Chegadas

ID	Companhia	Origem	Tipo	Tempo Operação	Tempo Circulação	Tempo Deslocacao
aviao2@danielpc	Lufthansa	São Paulo	carga	6	11	4
aviao4@danielpc	Delta	Braga	carga	4	10	6
aviao5@danielpc	Delta	Braga	carga	ATRASADO		
aviao8@danielpc	Lufthansa	Londres	carga	3	10	8
aviao7@danielpc	Lufthansa	São Paulo	comercial	ATRASADO		
aviao5@danielpc	Delta	Braga	carga	6	8	4
aviao6@danielpc	Delta	Dubai	carga	ATRASADO		
aviao7@danielpc	Lufthansa	São Paulo	comercial	7	11	3
aviao6@danielpc	Delta	Dubai	carga	3	11	5
aviao11@danielpc	Delta	Londres	carga	ATRASADO		
aviao10@danielpc	Delta	Londres	comercial	4	5	10
aviao15@danielpc	Delta	Dubai	comercial	ATRASADO		
aviao14@danielpc	American Airlines	Dubai	carga	ATRASADO		
aviao18@danielpc	Lufthansa	São Paulo	comercial	6	6	10
aviao16@danielpc	Delta	Braga	comercial	ATRASADO		
aviao11@danielpc	Delta	Londres	carga	3	14	9
aviao15@danielpc	Delta	Dubai	comercial	6	14	8
aviao14@danielpc	American Airlines	Dubai	carga	7	15	6
aviao16@danielpc	Delta	Braga	comercial	4	6	4
aviao20@danielpc	Emirates	Londres	carga	6	5	4

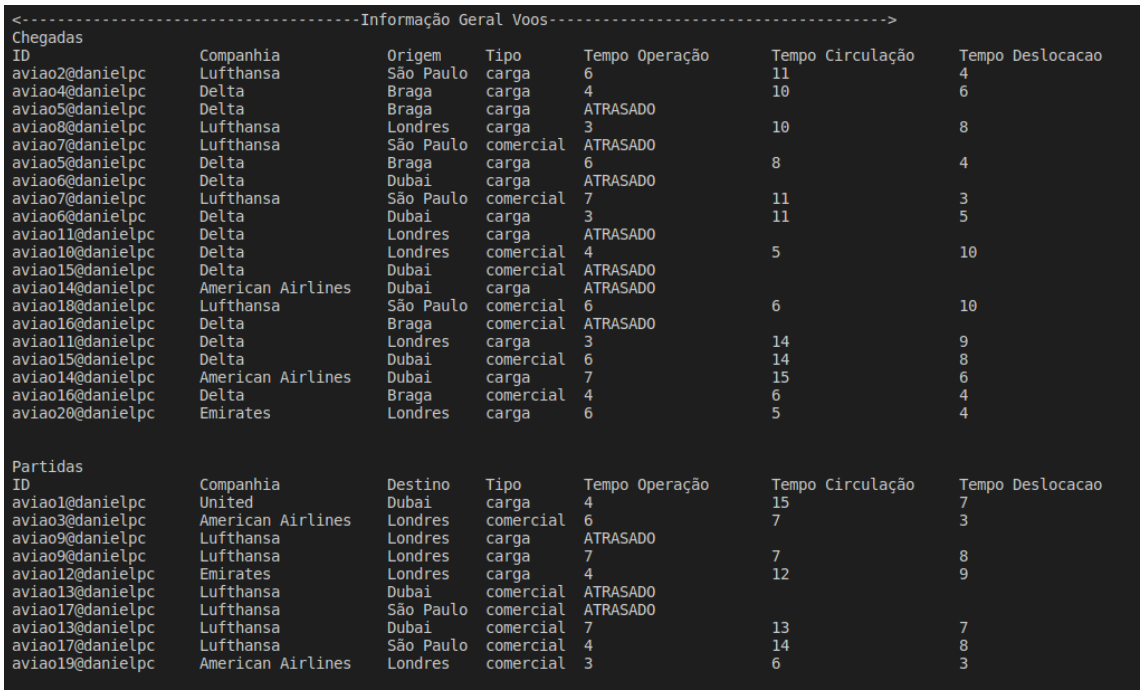
Partidas

ID	Companhia	Destino	Tipo	Tempo Operação	Tempo Circulação	Tempo Deslocacao
aviao1@danielpc	United	Dubai	carga	4	15	7
aviao3@danielpc	American Airlines	Londres	comercial	6	7	3
aviao9@danielpc	Lufthansa	Londres	carga	ATRASADO		
aviao9@danielpc	Lufthansa	Londres	carga	7	7	8
aviao12@danielpc	Emirates	Londres	carga	4	12	9
aviao13@danielpc	Lufthansa	Dubai	comercial	ATRASADO		
aviao17@danielpc	Lufthansa	São Paulo	comercial	ATRASADO		
aviao13@danielpc	Lufthansa	Dubai	comercial	7	13	7
aviao17@danielpc	Lufthansa	São Paulo	comercial	4	14	8
aviao19@danielpc	American Airlines	Londres	comercial	3	6	3

Figura 3.2: Teste com 20 aviões

Pelo que foi possível realizar a intenção, quer seja de descolar ou aterrar, de todos os aviões embora alguns realizaram a operação com atraso devido à indisponibilidade de gares ou pistas ou então devido a conflitos de prioridade.

De modo a forçar os aviões a irem para outro aeroporto, realizamos um testes apenas com gares comerciais pelo que os aviões de carga ao fim de 20 segundos sem resposta dirigem-se para outro aeroporto. O mesmo se aplica caso a fila de espera seja maior que 5.



<----------Informação Geral Voos---------->

Chegadas

ID	Companhia	Origem	Tipo	Tempo Operação	Tempo Circulação	Tempo Deslocacao
aviao2@danielpc	Lufthansa	São Paulo	carga	6	11	4
aviao4@danielpc	Delta	Braga	carga	4	10	6
aviao5@danielpc	Delta	Braga	carga	ATRASADO		
aviao8@danielpc	Lufthansa	Londres	carga	3	10	8
aviao7@danielpc	Lufthansa	São Paulo	comercial	ATRASADO		
aviao5@danielpc	Delta	Braga	carga	6	8	4
aviao6@danielpc	Delta	Dubai	carga	ATRASADO		
aviao7@danielpc	Lufthansa	São Paulo	comercial	7	11	3
aviao6@danielpc	Delta	Dubai	carga	3	11	5
aviao11@danielpc	Delta	Londres	carga	ATRASADO		
aviao10@danielpc	Delta	Londres	comercial	4	5	10
aviao15@danielpc	Delta	Dubai	comercial	ATRASADO		
aviao14@danielpc	American Airlines	Dubai	carga	ATRASADO		
aviao18@danielpc	Lufthansa	São Paulo	comercial	6	6	10
aviao16@danielpc	Delta	Braga	comercial	ATRASADO		
aviao11@danielpc	Delta	Londres	carga	3	14	9
aviao15@danielpc	Delta	Dubai	comercial	6	14	8
aviao14@danielpc	American Airlines	Dubai	carga	7	15	6
aviao16@danielpc	Delta	Braga	comercial	4	6	4
aviao20@danielpc	Emirates	Londres	carga	6	5	4

Partidas

ID	Companhia	Destino	Tipo	Tempo Operação	Tempo Circulação	Tempo Deslocacao
aviao1@danielpc	United	Dubai	carga	4	15	7
aviao3@danielpc	American Airlines	Londres	comercial	6	7	3
aviao9@danielpc	Lufthansa	Londres	carga	ATRASADO		
aviao9@danielpc	Lufthansa	Londres	carga	7	7	8
aviao12@danielpc	Emirates	Londres	carga	4	12	9
aviao13@danielpc	Lufthansa	Dubai	comercial	ATRASADO		
aviao17@danielpc	Lufthansa	São Paulo	comercial	ATRASADO		
aviao13@danielpc	Lufthansa	Dubai	comercial	7	13	7
aviao17@danielpc	Lufthansa	São Paulo	comercial	4	14	8
aviao19@danielpc	American Airlines	Londres	comercial	3	6	3

Figura 3.3: Teste mudança aeroporto

4 Conclusões e trabalho futuro

Na sua globalidade estamos bastante contentes com o trabalho desenvolvido e com a aplicação criada.

Pensamos ter atingido os objetivos propostos neste trabalho seguindo as técnicas e os padrões de desenvolvimento neste que foi o nosso primeiro contacto com sistemas multiagente, primordial à nossa formação enquanto Engenheiros Informáticos.

Este projeto permitiu-nos "meter as mãos na massa"e dados os seus requisitos operacionais e funcionais construir um sólido sistema.

Embora pensamos ter cumprido os objetivos propostos, em todos os projetos de grande dimensão e que envolvem bastante trabalho e rigor, há sempre pontos em que apresentamos mais dificuldades, e este projeto não foi exceção.

Pelo que os nossos principais pontos fracos e dificuldades recaíram sobre: o domínio da biblioteca SPADE e falta de tempo para a prévia modelação UML.

Todavia, apresentamos uma melhor facilidade a trabalhar como grupo, discutindo profusas vezes as ideias apresentadas mas mantendo o espírito de equipa como base do progresso do trabalho. Contudo, poderíamos estender o trabalho e implementar futuramente novas funcionalidades.