Enunciado do Trabalho Final - PCD 2012-2013 Controlador Aéreo

1 Objetivo

O trabalho deve fazer uma simulação de controlo aéreo que deve implementar obrigatoriamente:

- Sentinelas (de botões e do rato).
- Janelas, painéis, com desenho de imagens ou figuras e redefinição de (pelo menos) um método paintComponent().
- Concorrência de acesso a recursos partilhados (pelo menos um exemplo de acesso às células do espaço para a movimentação de aviões sem colisões).
- Possibilidade de ter algumas dezenas de processos ligeiros ativos em simultâneo a partilhar recursos.
- Possibilidade de resolução pelo utilizador de situações de impasse (*deadlock*) detetáveis visualmente.

O objectivo é simular uma aplicação/jogo de controlo aéreo. O espaço aéreo estará dividido numa quadrícula (de dimensão configurável) e poderá ter vários aeroportos (distribuídos aleatoriamente ou configuráveis em ficheiro).

Em cada aeroporto estará inicialmente estacionado (invisível) um conjunto de aviões.

Os aviões devem levantar voo e dirigir-se a um dos outros aeroportos, mas cada quadrícula apenas pode ser ocupada por um avião de cada vez. O utilizador deve resolver as situações de congestão / impasse de modo a que os aviões cheguem ao seu destino com a maior folga de combustível possível.

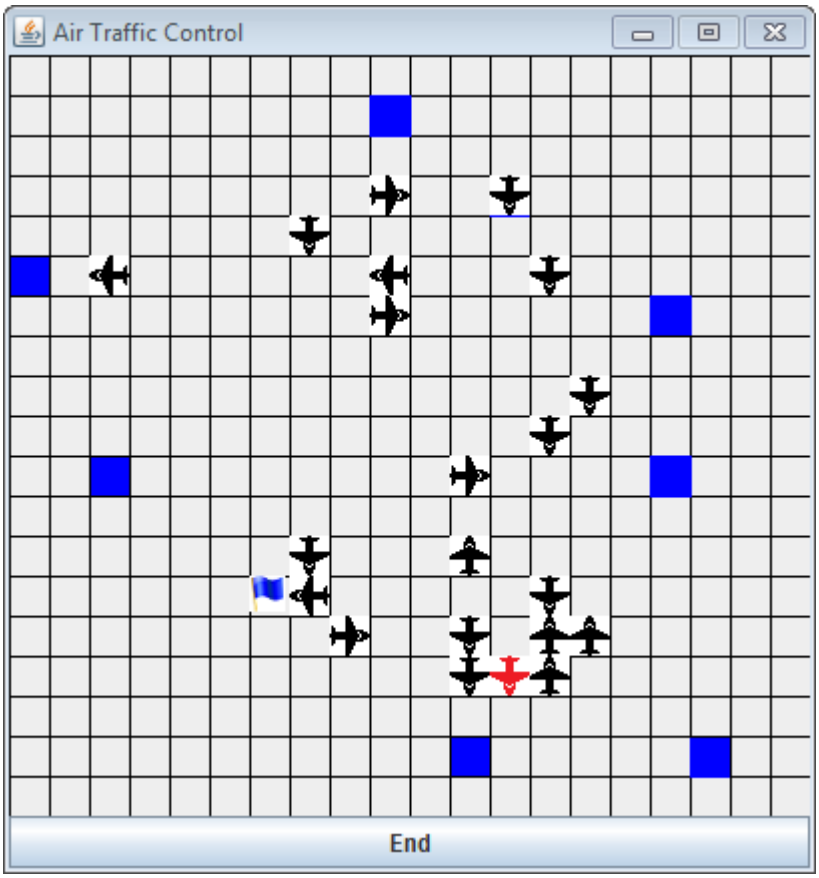

Figura 1. Exemplo de uma versão base da aplicação. Os quadrados azuis representam aeroportos, os aviões a preto encontram-se em movimento, o avião a vermelho está no limite do combustível. Em baixo à esquerda pode ver-se uma bandeira que marca a posição de um ponto de passagem a que um dos aviões se dirige. Este é apenas um exemplo, o interface gráfico não precisa de ter este aspeto, embora deva ser claro e fácil de usar.

2 Detalhes da aplicação

A aplicação pode ser feita numa de três versões. Recomenda-se que ponha primeiro a funcionar corretamente todas as funcionalidades da versão base antes de avançar para a versão completa local e só depois fazer a versão distribuída.

a Versão base

Deve ser possível ver uma grelha de dimensões configuráveis (o tamanho das células pode não ser configurável, é aconselhável que não seja inicialmente).

Caso exista um ficheiro de configuração com um nome predefinido (aeroportos.txt) este deve ser lido e devem ser criados aeroportos nas posições indicadas no ficheiro. Caso este não exista deve ser estabelecido um número de aeroportos por omissão e geradas aleatoriamente as posições dos aeroportos. Um aeroporto ocupa uma posição da grelha.

Deve ser criado um número aleatório de aviões em cada aeroporto e cada avião deve ter como destino um dos outros aeroportos. Cada avião é um processo ligeiro independente. Deve ser adicionado combustível ao avião que permita chegar ao destino com uma reserva de 10% a 20%.

Apenas a partir deste momento a aplicação deve ser mostrada ao utilizador e ser permitida a interação.

Cada avião deverá esperar um tempo aleatório, após o que deve iniciar as tentativas para descolar. Um avião pode descolar apenas quando a célula anexa ao aeroporto, na direção para onde se vai mover, está livre.

Em cada movimento o avião deve pedir acesso à célula seguinte e, caso esta esteja ocupada deve esperar (voando em círculos na célula onde se encontra) até que seja liberta. Um avião perde uma quantidade fixa de combustível por intervalo de tempo sempre que esteja no ar, quer esteja à espera da libertação de uma célula ou em movimento.

Caso o combustível no avião esteja abaixo de um limite (configurável) o facto deve ser assinalado visualmente (por exemplo, mudando a cor do avião para vermelho) de modo a que o operador/jogador tenha essa informação.

Deve ser possível criar para teste uma situação em que há apenas dois aeroportos, alinhados na mesma linha horizontal ou vertical e ambos têm pelo menos um avião que sairá em direção ao outro aeroporto pela rota mais curta, criando uma situação de impasse numa das células do caminho.

Deve ser possível ao utilizador selecionar um avião (usando o rato) e colocar uma bandeira que deve ser usada como ponto de passagem pelo avião. Isto permitirá ao operador definir rotas e resolver impasses quando estes forem detetados.

Ao aterrar no aeroporto de destino um avião escolhe novo aeroporto de destino e recomeça.

Cada rota completada por um avião deve dar 10 pontos + um ponto por cada intervalo de tempo que poderia ainda viajar com o combustível restante no porão ao aterrar. Cada queda de avião serão -100 pontos, cada bandeira colocada -1 ponto.

O objetivo é acumular o maior número de pontos num tempo limite.

b Versão completa local

Além de tudo o que consta da versão-base, a versão completa deverá ter uma ou todas as seguintes funcionalidades:

- Cada aeroporto deve ter um número fixo de pistas e permitir a descolagem simultânea de aviões até esse número. Caso haja uma aterragem em curso, não pode ocorrer nenhuma descolagem. Uma aterragem deverá demorar pelo menos 3 ciclos a completar.
- Cada aeroporto deverá gerar um número aleatório de passageiros (cada um, será um processo independente, eventualmente poderão ser agrupados em *ThreadPools*) em cada intervalo de tempo. Cada um destes passageiros será acrescentado à lista de passageiros para um destino (também gerado aleatoriamente). Cada avião deve ter um número limite de passageiros e ao aterrar, espera um número fixo de ciclos para reabastecer e depois, caso ainda não tenha destino, ser-lhe-á atribuído o destino que tiver o maior número de passageiros em lista de espera. Opcionalmente o utilizador poderá tomar controlo de um avião parado durante o reabastecimento e definir manualmente o destino de modo a escoar um aeroporto onde haja grande acumulação de passageiros. Cada passageiro que espere mais que um determinado tempo começará a produzir uma penalização contínua de 1 ponto por cada intervalo de tempo (configurável).
- É útil para o operador poder detetar rapidamente: a direção em que se dirige cada aparelho, o seu estado de consumo de combustível, o seu estado de movimento (em espera por uma célula ou em movimento livre) se está selecionado para lhe ser indicado um ponto de passagem, se está a caminho do destino ou de um ponto de passagem, se um aeroporto está muito congestionado ou não, etc. Será útil descobrir boas soluções gráficas para mostrar tanta informação quanta for possível sem tornar a janela ilegível. Um bom uso das cores e o aparecimento de alguns destes detalhes ao passar o rato sobre um avião/aeroporto são soluções a considerar.
- É útil também ter um mostrador para a pontuação que receba informações concorrentemente de todos os elementos que podem influenciar as pontuações.

C Versão distribuída, para vários utilizadores.

A terceira versão é para vários utilizadores jogarem de uma forma distribuída. A aplicação deve ter um servidor a correr numa máquina e porto conhecidos. Os clientes são as aplicações onde cada jogador participa no jogo. As regras do jogo distribuído deve ser ligeiramente diferentes:

- Cada jogador deve ter um conjunto predefinido de aviões.
- O objectivo é transportar o máximo de passageiros entre os aeroportos.
- Os jogadores competem pelos passageiros.
- Todas as restantes regras são idênticas às versões anteriores.
- Deve ser possível jogar a versão de apenas um jogador (local) ou a versão distribuída para vários jogadores.

3 Documentos

Será necessário produzir os seguintes documentos:

- Um relatório onde se expliquem as soluções de sincronização adoptadas face aos problemas de concorrência detectados. Este relatório deve ter no máximo duas páginas com letra do tipo *arial* de tamanho 11.
- Um manual de utilização que deve ser explícito e sucinto com um tamanho máximo de uma página, com letra de tamanho 11 do tipo *arial*.

4 Entrega

O número de elementos por grupo de trabalho deve ser dois.

O trabalho deve ser arquivado usando Export/Archive File, e ser **entregue no e-learning e enviado por mail para o docente** que dá a aula prática em que está inscrito.

A **data limite** de entrega é 14/12/2012 às 24h.

Penalizações**:** 1 valor por cada dia de atraso. Não serão aceites trabalhos a partir das 24h de dia 16/12/2012.

5 Avaliação

A versão base é cotada de 0 a 12 valores (em 20), sendo que apenas notas superiores a 8 valores permitem o acesso à frequência e a aprovação à UC.

A versão completa local terá notas de 0 a 16.

A versão distribuída incluindo versão completa terá notas de 0 a 20.

A avaliação do Trabalho Final terá em conta:

- a correcta execução do programa de acordo com as especificações presentes neste enunciado e demais opções tomadas pelo grupo, desde que justificadas no relatório do trabalho;
- a correcta estruturação das classes e métodos que compõem o programa;
- a correcta utilização dos recursos partilhados bem como a correcta e eficiente sincronização dos diferentes participantes;
- a robustez da execução relativamente à ocorrência de excepções ou situações de erro provocadas pelo utilizador;
- a correcta formulação do manual de utilização e relatório do trabalho, incluindo aspectos como o poder de síntese, a clareza e a correcção gramatical e ortográfica do português;
- a criatividade do jogo.

A discussão do trabalho será na semana de 17/12/2012.

6 Prémios

Serão atribuídos os seguintes certificados aos melhores projetos:

1. Melhor Projecto
2. Melhor Interface Gráfica
3. Prémio de Criatividade