Universidade do Porto
Faculdade de Engenharia
FEUP

Jogo de Estratégia RISK

Relatório Final

4º ano do Mestrado Integrado em Engenharia Informática e Computação

Elementos do Grupo:

André Humberto Trigo de Bordalo Morais – 200702669 – ei07122

João Alberto Trigo de Bordalo Morais – 201208217 – ei12040

13 de Dezembro de 2015

Índice

Objetivo

● Descrição do cenário

Este trabalho está a ser realizado no âmbito da unidade curricular de Agentes e Inteligência Artificial Distribuída e consiste na implementação do jogo de tabuleiro RISK.

O RISK é um jogo de estratégia por turnos que pode ser jogado por 2 a 6 jogadores num tabuleiro que representa (na versão mais conhecida) o mapa politico da Terra, dividido em 42 territórios e agrupados em 6 continentes.

O objectivo do jogo é conquistar todos os territórios e, consequentemente, eliminar todos os jogadores.

Com esta finalidade os jogadores tentam capturar territórios inimigos com os seus exércitos, travando batalhas.

Cada turno é composto por 3 fases : Obter e distribuir tropas, Atacar (por lançamento de dados) , reforçar a posição.

É permitida a comunicação entre os jogadores no desenrolar do jogo com vista à formação de alianças temporárias que podem ou não ser traídas.

● Objectivos do trabalho

O objectivo deste trabalho é especificar e implementar um sistema distribuído e descentralizado influenciado pelo paradigma de sistema de multi-agente. Num contexto do mapa político do RISK, pretende-se com este trabalho implementar agentes de vários tipos: aleatórios, estratégicos e ainda BDI (*Beliefs*, *Desires*, *Intentions*). Tal como é descrito nas regras de jogo, os jogadores poderão criar alianças, com o intuito posterior de quebrar ou não. Para tal é necessário haver comunicação entre agentes e ainda a aplicação de técnicas

de negociação e modelos de confiança. O objectivo final é compreender qual das estratégias é a mais eficaz. Salienta-se que neste jogo está associado um factor sorte, deste modo, só após vários jogos é que é possível tirar conclusões.

Em suma, é nosso objectivo implementar multi-agentes com diferentes estratégias a comunicarem entre si procurando estabelecer alianças, aplicando técnicas de negociação e modelo de confiança. Para além disto haverá o contacto Agente ⇔ Ambiente (Tabuleiro) onde será ditado o fluxo do jogo.

Especificação

● Identificação e caracterização dos agentes

O tipo de agentes implementado é do tipo reactivo.
No inicio do seu turno este retira as suas percepções do ambiente sob a forma de três tipos : *Reinforce Perception , Attack Perception e Fortify Perception.*

- ***Reinforce Perception***

  São recolhidos todos os territórios pertencentes ao agente em questão assim como o numero de tropas disponíveis para o efeito,

- ***Attack Perception***

São recolhidos do ambiente todos os pares território de origem / território alvo (território adjacente de ocupação adversaria), em que o número de tropas no território de origem seja superior

- ***Fortify Perception***

São recolhidos todos os pares territórios de origem / território destino (obrigatória a existência de ligação entre os dois) Porém, não está implementada a 100%.

A próxima fase consiste em gerar uma *Reinforce Action* , *Attack Actions* e *Fortify Action* a partir das percepções disponíveis.

- ***Reinforce Action***

  É escolhido o território menos guarnecido de tropas para ser alvo de alocação.

- ***Attack Action***

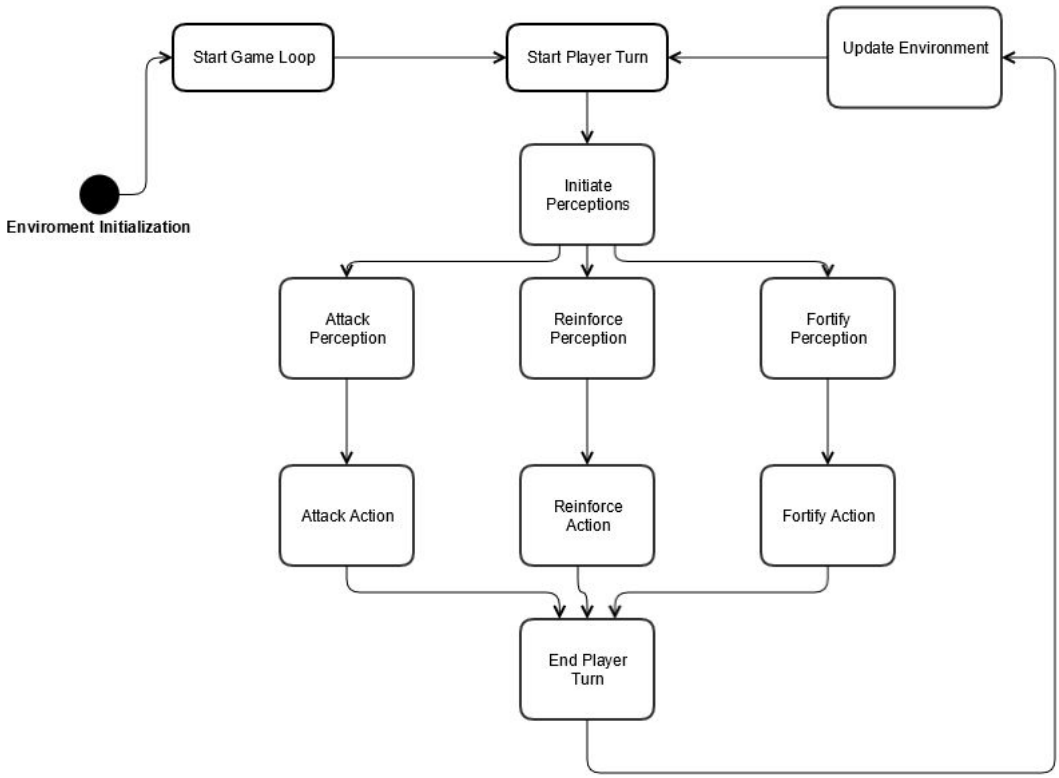
  É tomada a acção de tentar conquistar o território em que a diferença de soldados é mais acentuada.

- ***Fortify Action***

  Analisando as *Fortify Perception*, o agente escolherá o território em maior perigo eminente, isto é, o agente escolherá dos seus territórios aquele em que na sua adjacência tem exércitos inimigos em maior número.

- **Protocolos de interação**

Agente⇔Ambiente O agente retira a informação relevante do ambiente por intermédio das suas *perceptions* e impõe as suas modificações ao ambiente por meio de *actions*.



Desenvolvimento

● Plataforma/Ferramenta

O trabalho foi desenvolvido em java, recorrendo ao IDE Eclipse-Luna, nos sistemas operativos Win8.1 e Win7. Por cima disto, foi utilizada a ferramenta *jadex* para criar o agente e ambiente onde o jogo se desenrola.

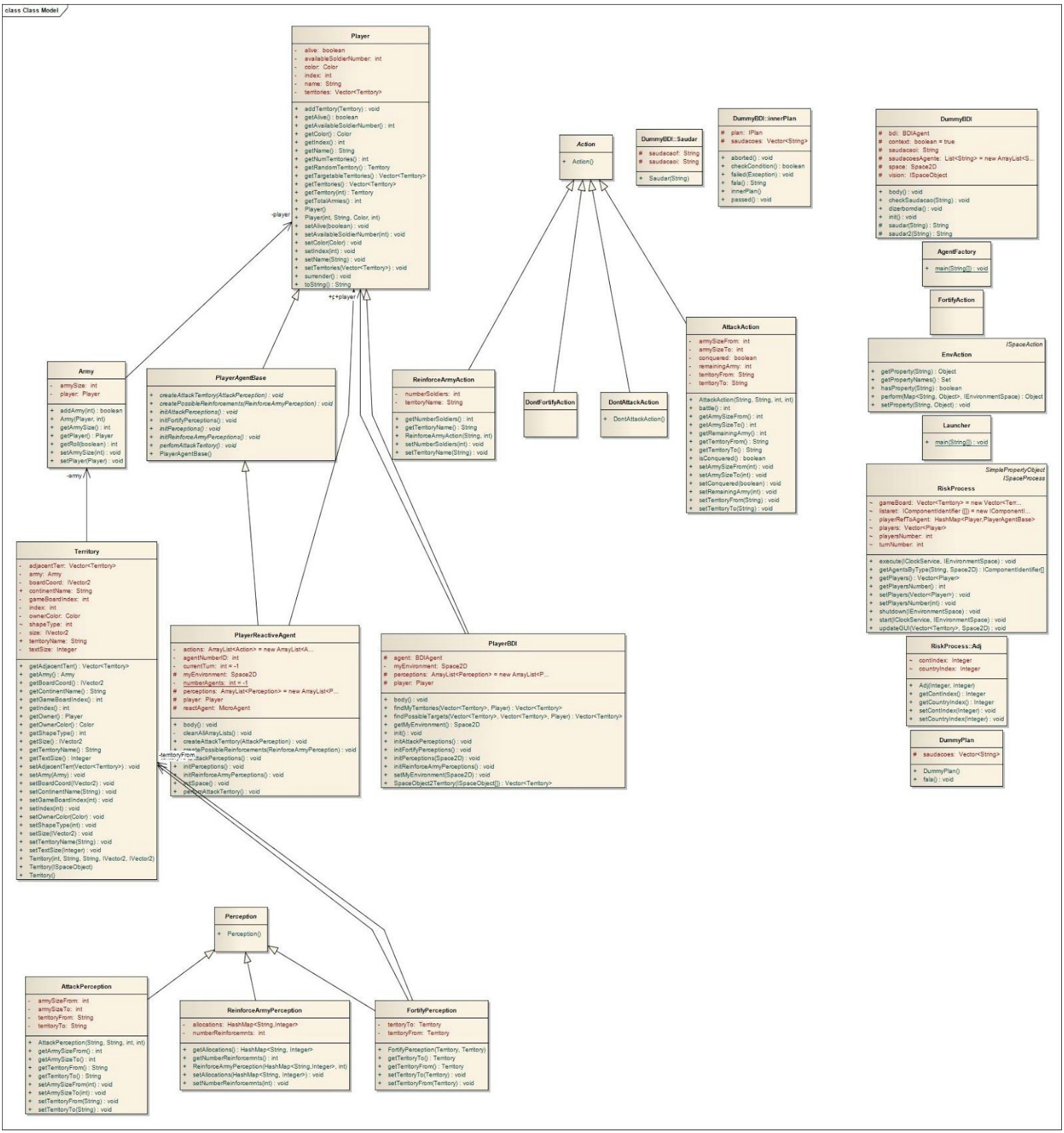
● Estrutura da aplicação, módulos, diagrama de classes

A aplicação está dividida em 5 pacotes(*packages*):

- **actions:** ações possíveis dos agentes
  - Action: classe template das ações.
  - AttackAction: classe usada para representar a ação de ataque do agente.
  - DontAttackAction: classe usada para representar a retirada de ataque do agente (útil para o *Random Agent* - não implementado - e para o *Deliberative Agent* - não implementado. Neste último importante para ele reavaliar as opções e, se pertinente, optar por outras ações).
  - DontFortifyAction: classe usada para representar a nenhuma necessidade de o agente fortificar um determinado território.
  - FortifyAction: classe usada para representar a ação de o agente fortificar um determinado território seu.
  - ReinforceArmyAction: classe usada para representar a ação de em que territórios o agente escolheu para apoiar no inicio da sua jogada.

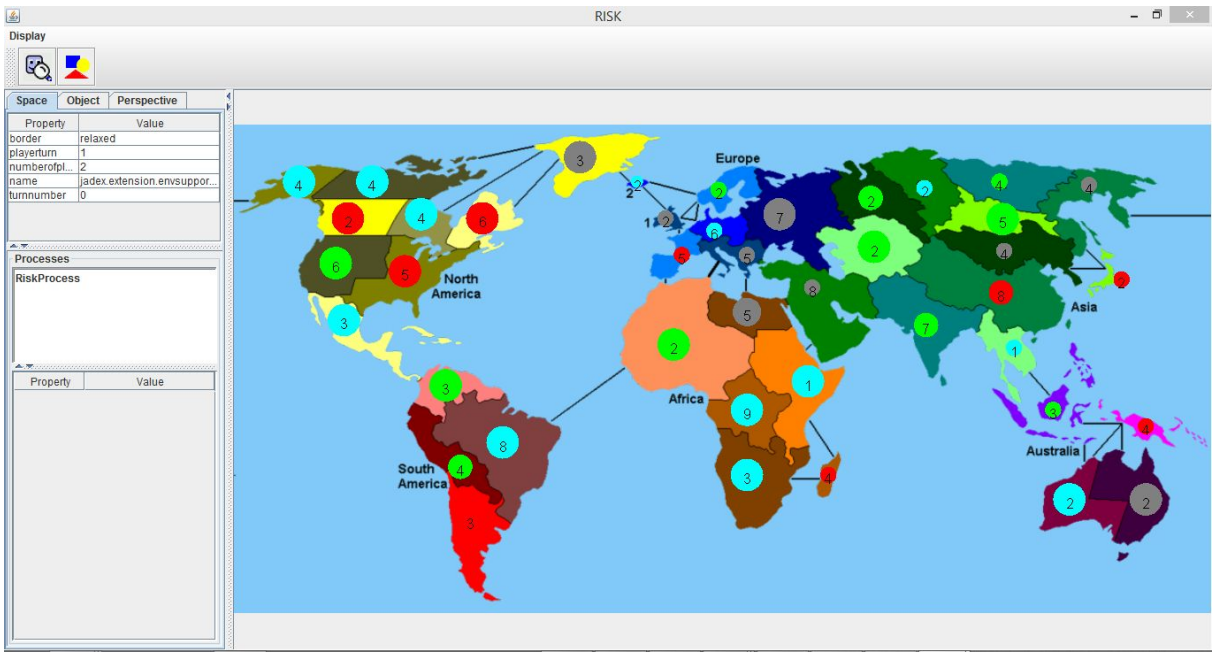
- **agents:**
  - DummyBDI: classe de teste para experimentar o conceito BDI. Não foi utilizado em jogo.
  - PlayerAgentBase: classe template base com os métodos base e transversais ao agentes.
  - PlayerReactiveAgent: classe do agente reativo.
- **game:**
  - Army: Esta classe permite representar um exercito, seja o jogador a que pertencem e a quantidade associada.
  - Launcher: Esta classe permite definir a configuração de agentes a ser utilizada.
  - Player: classe que representa um jogador do jogo , detém as informações relativas aos territórios de um jogador assim como o numero de tropas disponível.
  - RiskProcess: processo principal do jogo : a sua função de start inicia o ambiente e faz a configuração das condições de jogo, a sua função de execute contem a lógica de turnos associada ao jogo e contem métodos para actualizar o seu estado.
  - Territory: classe representativa de um território de jogo , no qual estão referenciadas informações relativas ao exercito que o ocupa e tambem em relação aos territórios adjacentes.
- **perceptions:**
  - AttackPerception: classe utilizada para captar do ambiente os territórios possíveis de atacar.
  - FortifyPerception: classe utilizada para ver quais os territórios. mais vulneráveis - não implmentada.
  - Perception: classe template das percepções.
  - ReinforcementPerception: classe que permite captar do ambiente quais os territórios possíveis de receber apoio de tropas.
- **plans:**
  - DummyPlan: classe de plano do agente DummyBDI

E com o seguinte diagrama de classes:



● Detalhes Relevantes da Implentação

De modo a ver a atividade dos agentes criados, é apresentada a seguinte Interface gráfica:



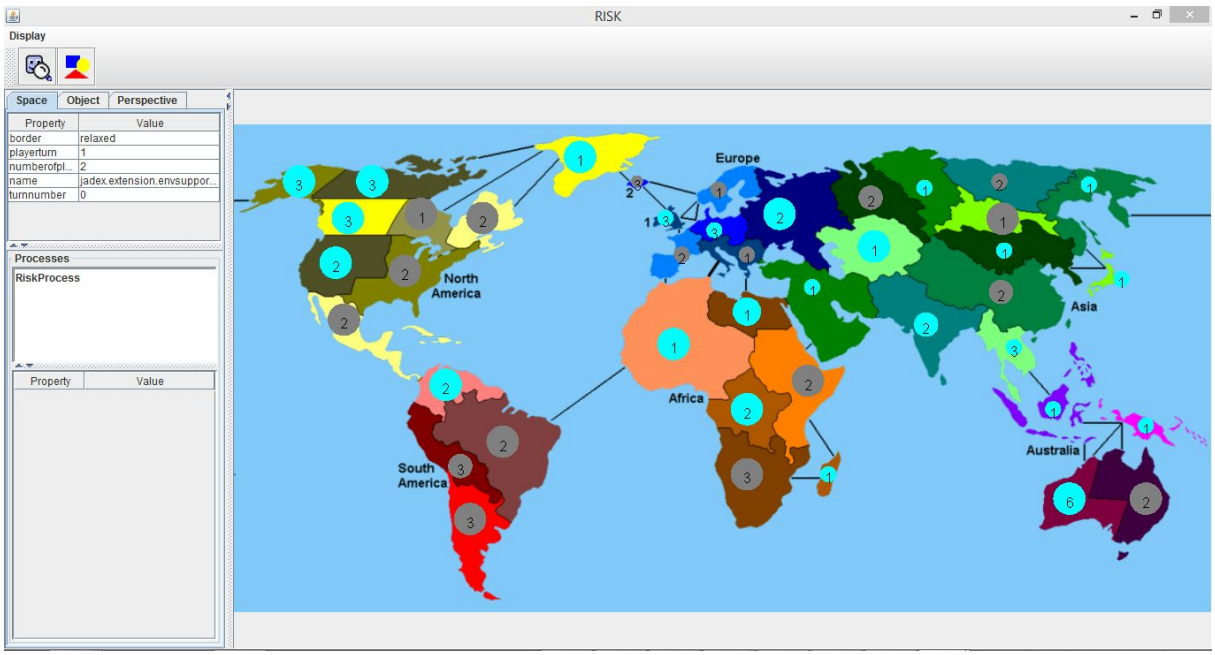
Nela é visível o mapa-político do jogo Risk. Nesta instância do jogo, é possível ver os 5 continentes, os respectivos territórios com um círculo com uma de 4 cores diferentes, representando 4 jogadores diferentes (*Cyano*, Vermelho,Verde e Cinzento). Dentro do círculo tem um número representando de tropas presentes nesse território. No painel do lado esquerdo é possível ver informações relevantes sobre o mapa e o território seleccionado.

Experiencias

• Objectivo de cada experiência

Experiência 1: Pegar em dois *Reactive Agents* e colocá-los a jogar e determinar qual é o vencedor.

Estado inicial da experiência:



• Resultados

Experiência 1: experiência sem êxito devido à concorrência entre os agentes a jogar, impossibilitando a continuidade dos mesmo jogarem. Verifica-se apenas a execução de uma jogada por cada Agente, isto é, introdução de exército e conquista de um território inimigo.

Conclusões

- **Análise dos resultados das experiências levadas a cabo**

Na teoria, o agente do tipo BDI deveria ser o agente com maiores hipóteses de vitória relativamente a agentes do tipo Agressivo, Reativo, Deliberatido, e Aleatório.

Na prática tal não é possível de provar uma vez que o agente Reativo é o único implementado.

Relativamente à experiência conduzida apenas pode-se constatar que os agentes interagem com o ambiente.

- **Desenvolvimento do trabalho e aplicabilidade de SMA ao cenário proposto**

A aplicabilidade de Sistemas Multi-Agentes tem imensas potencialidades na vida real. Aplicada a jogos, permite criar oponentes ao utilizador, não humanos, com inteligência quase própria, isto no sentido individual do agente. Escalando para vários agentes, permite avaliar a competição entre eles de modo a atingir a vitória.

No contexto da Unidade Curricular, permite consolidar e por em prática vários conceitos, nomeadamente os vários conceitos de Agente (Reativo, deliberativo, BDI, etc..), Multi-Agente, Ambiente, OBjetivo(*Goal*); as arquiteturas dos vários agentes, comunicação entre os agentes.

No entanto, não pudemos deixar de expressar o nosso descontentamento com a documentação existente. Recentemente a ferramenta *jadex* sofreu atualizações o que conduz a pouca documentação existentes e termos de

aprender por versões anteriores e adaptar para a atual versão. Neste sentido, a limitação na documentação, por consequência, dificultou-nos na elaboração do projeto.

Deste modo, o projeto ficou muito distante das nossas espectativas.

Melhoramentos

Relativamente ao que ficou concretizado no projeto, falta melhorar a concorrência entre agentes com o objetivo de prosseguir com o fluxo de jogo.

Relativamente ao esperado, um dos pontos a melhorar será implementar maior diversidade de Agentes, exemplo do agente Deliberativo, BDI, estratégia Agressiva, entre outros; outro melhoramento importante será implementar comunicação entre agentes de modo a estabelecer alianças ou pactos; bem como, aliado às alianças, o grau de confiança destes pactos. Para estabelecer a comunicação entre agentes será necessário implementar serviços.

Recursos

Até agora, os recursos que temos utilizado dividem-se em dois tipos:

- **Bibliográficos:**

Introduction to MultiAgent Systems - M.WooldRidge, John Wiley &Sons

- **Links**

https://github.com/joaoBordalo/feup-AIAD.git-repositorio git do projecto

www.hasbro.com/common/instruct/risk.pdf - descrição do jogo RISK

http://www.totaldiplomacy.com/ - descrição detalhada das várias estratégias de jogo dependendo da situação.

https://en.wikipedia.org/wiki/Subsumption_architecture - definição e especificação de agentes reativos

https://paginas.fe.up.pt/~eol/AIAD/1516/aulas/AIAD1_2abc.pdf - definição e especificação de agentes
https://paginas.fe.up.pt/~eol/AIAD/1516/aulas/AIAD2d.pdf - agentes BDI

- **Software:**

https://sourceforge/projects/jadex - Jadex Active Components download
https://activecomponents.org - pagina oficial da ferramenta Jadex
https://paginas.fe.up.pt/~eol/AIAD/jadex/doku.php?id=agentes - tutorial sobre Jadex

- **Esforço Percentual dos Elementos do grupo:**

André Humberto Trigo de Bordalo Morais: 50%

João Alberto Trigo de Bordalo Morais: 50%

Manual de Utilização

Para correr o programa basta dentro do IDE, nomeadamente Eclipse, correr a classe ‘Launcher’ presente no package ‘game’.