



ISEL
INSTITUTO SUPERIOR DE
ENGENHARIA DE LISBOA

Licenciatura em Engenharia
Informática e Multimédia
(LEIM)

Modelação e Simulação de Sistemas Naturais

2020/2021 – Semestre Inverno

Projeto Final



LEIM33D – 24 fevereiro 2021

Docente: Arnaldo Abrantes

Paulo Jorge nº 45121

Luís Fonseca nº 45125

Índice

| | |
|--|----|
| Introdução..... | 3 |
| A história. | 3 |
| Conceitos Teóricos | 4 |
| Componente Prática usando o Anylogic: | 6 |
| Componente Prática usando o Java Processing: | 8 |
| Diagrama UML | 10 |
| Resultados..... | 13 |
| Conclusões | 16 |
| Bibliografia | 17 |

Índice de Figuras

| | |
|--|----|
| Figura 1 - diagrama de Stocks e Flows criado a partir do AnyLogic..... | 6 |
| Figura 2 - diagrama UML do agente autónomo..... | 10 |
| Figura 3 - diagrama UML do autómato celular | 11 |
| Figura 4 - diagrama UML do ecossistema | 12 |
| Figura 5 - diagrama UML do menu | 12 |
| Figura 6 - resultado do ambiente aquático desenvolvido | 13 |
| Figura 7 - resultado do ambiente rural, com intervenção do caçador | 14 |
| Figura 8 - resultado do comércio de carne e pele | 14 |
| Figura 9 - número de recursos obtidos..... | 15 |
| Figura 10 - salvar espécies | 15 |
| Figura 11 - informação acerca do número de espécies vivas | 15 |

Introdução

No âmbito do desenvolvimento do projeto final da unidade curricular de Modelação e Simulação de Sistemas Naturais é de extrema importância referir e destacar o papel que a simulação computacional teve e tem no estudo de sistemas complexos.

O sistema desenvolvido no projeto final da unidade curricular pretende simular um ecossistema terrestre, composto por 6 espécies, sendo três delas presas (Gazela, Cegonha e Peixe), três predadores (Caçador, Leão e Crocodilo) e um meio terrestre aéreo. As presas vão-se alimentar de algas e plantas, que é o recurso renovável que tem de ser gerido, enquanto os predadores se alimentam de carne, neste caso, o leão alimenta-se de gazelas e caçadores, enquanto os crocodilos se alimentam de cegonhas. Na criação deste ecossistema procurou-se encontrar os valores que apresentassem equilíbrio, havendo uma certa procura da sustentabilidade, de modo que todas as espécies possam prosperar, não ocorrendo nenhuma extinção. O projeto foi desenvolvido deste modo por apresentar aspetos de simulação realizados sobre um sistema natural que apresenta interesse pela sua diversidade.

A história.

A nossa história, passa-se no presente planeta Terra, algures no vasto solo africano, da savana.

Assim de um modo muito simplificado, nós criamos um pequeno ecossistema, onde plantas através da fotossíntese (não implementada, simulada apenas com valores arbitrários), surgem servindo como alimento a diversas cadeias alimentares. Portanto através da alimentação dos seres, competição por recursos, nomeadamente alimento e simulação da reprodução dos seres, onde aqueles que mais comem, mais se reproduzem, iremos explicar como funciona um ecossistema, muito simples.

Conceitos Teóricos

Sustentabilidade

A sustentabilidade é um processo que permite o desenvolvimento da geração atual sem prejudicar as necessidades das próximas gerações. Isto é, é necessário que os recursos naturais, os quais podem ou não ser finitos, sejam utilizados de forma inteligente, de forma que os mesmos se mantenham no futuro, garantindo assim o desenvolvimento sustentável. Assim sendo, de forma a garantir, o desenvolvimento sustentável as ações que afetam os nossos recursos naturais devem possuir uma abordagem sistematizada, tendo em conta a proteção do ambiente.

Consumidores e produtores

Os **consumidores** são representados por todos os seres vivos que consomem algo, ou seja, que comem algo, como por exemplo, os animais. Estes, consomem os alimentos fornecidos pelas plantas ou por outros animais. Os **produtores** são os seres vivos que fabricam o seu próprio alimento, no seu interior e, por conseguinte, não comem outras plantas ou animais, como por exemplo, as plantas. Os animais também podem ser produtores, uma vez que podem servir de alimento para outros animais. Neste projeto, as algas são consideradas produtores e os consumidores de algas. Entretanto, os peixes e as cegonhas também serão produtores, uma vez que servem de alimento para os crocodilos, respetivamente, sendo por isso estes dois últimos, consumidores.

Terreno

O terreno consiste num autómato celular, e no qual, é representado por uma grelha 2D, formada por células que possuem um estado. Cada célula escolhe aleatoriamente o seu estado e aplica, iterativamente, a regra da maioria com raio unitário e vizinhança de Moore, até que o autómato celular se dirija para uma configuração final. Neste projeto o terreno possui quatro estados, os quais são designados por:

1. **Empty**: células que representam o vazio;
2. **Food**: células que representam o alimento;
3. **Fertile**: células que representam a parte fértil do terreno, ou seja, que podem produzir alimento;
4. **Obstacle**: células que representam obstáculos.

No decorrer da simulação, uma célula com alimento passará a fértil sempre que uma presa estiver sobre ela e consumir o alimento. Nesse caso, a célula retornará ao estado inicial (alimento) ao fim de um determinado tempo aleatório (regeneração do terreno). Esta regeneração varia entre 5 e 18+5 segundos.

Agentes Autónomos

Um agente inteligente é aquele que atua por si próprio de forma independente e sem intervenção de outros sistemas. Ou seja, um agente inteligente é também um agente autónomo uma vez que a autonomia é uma característica intrínseca à inteligência. Este projeto é constituído por dois tipos de agentes autónomos, sendo estes as presas e os predadores.

As **presas** e os **predadores** são agentes que vagueiam pelo terreno em busca de alimento. Cada um deles, possui uma energia, a qual vai sendo consumida 1 unidade por segundo, de forma constante, com o passar do tempo, devido ao seu metabolismo. Entretanto, se o agente não conseguir evitar o obstáculo e passar por ele, a sua energia é consumida de forma acelerada.

Ambos os agentes, obtém energia ao se alimentarem, no caso das presas, sendo estas representadas pelos peixes e cegonhas, ganham energia ao se alimentarem das algas, e os predadores os leões e os crocodilos, ganham energia ao se alimentarem das gazelas e das cegonhas respetivamente. Desta forma, as presas morrem quando são capturadas pelo seu predador e ambos os agentes morrem de causa natural, através de um modelo determinístico, ou seja, quando a sua energia se esgota (valor igual a 0). Ambos os agentes se reproduzem de forma assexuada (determinística), ou seja, quando a sua energia é superior a 200. Nesse processo, a energia do pai é dividida em partes iguais entre o mesmo e o seu filho.

Componente Prática usando o Anylogic:

Durante a realização do projeto, o grupo elaborou um modelo no Anylogic, para eventuais estudos e retirar conclusões, tanto das taxas da natalidade, como de mortalidade, das duas espécies de presas e predadores em estudo. O modelo pode ser visto na figura em baixo:

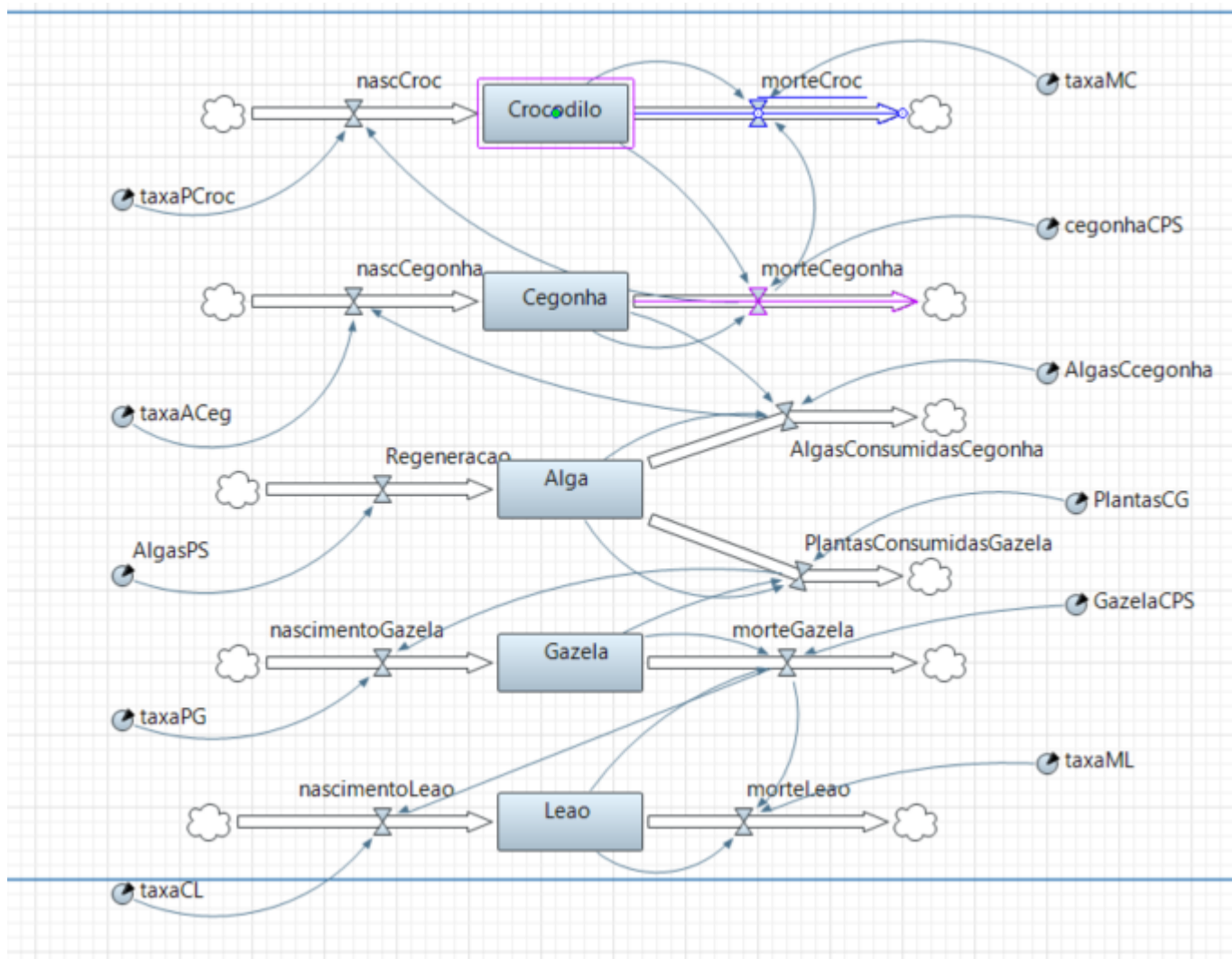


Figura 1 - diagrama de Stocks e Flows criado a partir do AnyLogic

O sistema é constituído pelos seguintes stocks:

- **Leao:** corresponde a uma população de tubarões;
- **Gazelas:** corresponde a uma população de peixes;
- **Cegonhas:** corresponde a uma população de camarões;
- **Crocodilos:** corresponde a uma população de baleias;
- **AlimentVegetal(algas):** representa o tipo de terreno que vai servir de alimento para as cegonhas e gazelas;

O sistema é constituído pelos seguintes flows:

- ***nascCegonha***: representa o nascimento da população de cegonhas;
- ***nascimentoGazelas***: representa o nascimento da população de gazelas;
- ***nascLeao***: representa o nascimento da população de leões;
- ***nascimentoCroc***: representa o nascimento da população de crocodilos;
- ***Regeneracao***: representa a taxa de regeneração das algas por segundos;
- ***AlgasConsumidasCegonha/PlantasConsumidasGazela***: representa as algas e plantas consumidas das gazelas e das cegonhas;
- ***morteLeao***: representa o número de mortes da população de leões;
- ***morteGazela***: representa o número de mortes da população de gazela;
- ***morteCroc***: representa o número de mortes da população de crocodilos;
- ***morteCegonha***: representa o número de mortes da população de cegonhas;

O sistema é constituído pelas seguintes variáveis:

- ***taxaPCroc***: consiste na taxa de natalidade dos crocodilos;
- ***taxaACeg***: consiste na taxa de natalidade das cegonhas;
- ***AlgasPS***: consiste no aparecimento de algas por segundo;
- ***taxaPG***: consiste na taxa de natalidade das cegonhas;
- ***taxaCL***: consiste na taxa de natalidade dos leões;
- ***taxaMC***: consiste na taxa de mortalidade dos crocodilos;
- ***cegonhaCPS***: gazelas consumidas por segundo;
- ***AlgasCcegonha***: consiste nas algas consumidas pela cegonha num segundo;
- ***PlantasCG***: consiste nas plantas consumidas pela gazela num segundo;
- ***GazelaCPS***: gazelas consumidas por segundo;
- ***taxaML***: consiste na taxa de mortalidade dos leões;

Componente Prática usando o Java | Processing:

Em termos de componente prática relacionada com Java, foram criadas várias classes que permitiram a realização deste projeto. Assim sendo, algumas classes foram fornecidas pelos docentes da disciplina, pelo que, a nossa explicação irá incidir na explicação das novas classes que geramos.

Package aa(agentes autómatos)

Acerca deste package, temos as classes:

- **Boid:** representa um agente autónomo;
- **BoidDNA:** representa o ADN do boid. Funciona como uma “base de dados”, no qual guarda os valores da força e da visão;
- **BoidOverGrid:** permite que o boid esteja por cima da grelha. Esta grelha corresponde a um autómato celular que foi usada para o desenho do nosso terreno;
- **Flock:** representa um conjunto de boids, que se encontram em grupo, vagueando pelo mundo;
- **FlockOverGrid:** permite que o flock esteja por cima da grelha. Esta grelha corresponde a um autómato celular que foi usada para o desenho do nosso terreno;
- **Mover:** permite que qualquer agente possa mover, aplicando forças por vetores.

Para a classe *BoidOverGrid* realizamos métodos que permitem que o Boid evite obstáculos. Os métodos criados foram: *hasObstacle(Minefield t)* e *avoidObstacle(MineField t)*. O primeiro método permite verificar que existe um obstáculo no terreno, já o outro permite evitar um obstáculo, se estiver no campo de visão do Boid.

Para a classe *flockOverGrid* criamos um flock, com uma instancia de *BoidOverGrid*, e no final o método *applyBehaviour(Boid b, Minefield t)* que permite aplicar diferentes comportamentos em qualquer agente.

Package ca (autómato celular)

Contem todas as classes criadas para um autómato celular. Foi através deste conceito que foi possível a construção do nosso terreno. As classes criadas foram:

- **Cell:** corresponde a uma célula na nossa grelha do terreno;
- **CellularAutomata:** permite fazer o desenho da nossa grelha e permite aplicar vizinhança.
- **GridConstants:** grelha onde estão as variáveis que permitem o desenho da nossa grelha;
- **LifeCell:** corresponde ao tempo de vida de uma célula;
- **MajorityCA:** permite aplicar vizinhança;
- **MajorityCell:** permite aplicar a regra da maioria;
- **Minefield:** permite aplicar obstáculos no nosso terreno, conseguindo assim os nossos agentes evitarem esses obstáculos;

Foi na classe *MajorityCell* que aplicamos a regra da maioria, com o método *applyMajorityRule()*.

Foi na classe *MineField* que aplicamos obstáculos na grelha, com o método *setObstacles()*.

É na classe *CellularAutomata* que fazemos a distinção dos diferentes estados (Água, rural, etc). Visto que a implementação de cada ambiente é semelhante, apenas iremos explicar como procedemos para aplicar ambiente terrestre. Para desenhar ambiente terrestre, criamos o método *setGrassStates()* que permite aplicar relva no nosso ambiente. Para fazermos o desenho da relva, e distinguir entre água e relva, fomos procurar os diferentes valores das células (os valores foram feitos em tentativa erro, até o grupo ficar satisfeito com a quantidade de terreno que colocou). Para cada linha e coluna da grelha, usamos o método *setState* que aplica um tipo de estado, neste caso, aplicamos o estado “Grass”, ou seja, erva.

Package Ecosystem

É neste package que temos todas as classes para a realização do nosso ecossistema.

As classes usadas foram:

- **Animal:** corresponde a uma espécie, no qual nasce, morre, reproduz e evita obstáculos;
- **Cacador:** corresponde a um predador, que caca gazelas. Quando mais alimento caçar, mais consegue aumentar o número de mercados produzidos;
- **Cegonha:** presa que se alimenta de algas, no meio aquático;
- **Crocodilo:** predador que se alimenta de cegonhas, no meio aquático;
- **Fish:** presa que se alimenta de algas no meio aquático;
- **Gazela:** presa que se alimenta de plantas, que surgem com o decorrer do tempo, no ambiente terrestre;
- **Leao:** predador que se alimenta de caçadores e de gazelas.
- **Plane:** funciona como uma funcionalidade ao utilizador, permitindo mover para onde quiser, consoante o clique do rato;
- **Animal:** corresponde a uma espécie, no qual nasce, morre, reproduz e evita obstáculos;
- **Population:** representa o conjunto duma população em geral, neste caso, de todas as espécies que surgem no ecossistema;
- **RuralManager:** permite controlar o ambiente rural, ou seja, adicionar terreno fértil, assim como mercados, lendo as diferentes células da grelha2D, criada pelos autómatos celulares;
- **Mundo:** é onde é criado o nosso mundo, aplicando os diferentes estados. Cada célula representa um diferente estado no ecossistema, desde água, até terreno terrestre;
- **MundoConstantes:** funciona como uma “base de dados”, permitindo alterar as taxas de natalidade, de mortalidade, a quantidade de presas e predadores e a cor de ambas as espécies;
- **Patch:** permite que as células regenerem. A célula muda de estado consoante foi comida ou existe alimento;
- **Terreno:** autómato celular que contem uma grelha. Cada célula corresponde a 4 estados: alimento, água, terreno fértil e bloco de areia;

Na classe “RuralManager” é de importante salientar a criação do método *buildMarket(Terrain terrain)*. Este método permite construir um mercado, para a venda de pele e de carne. Dentro deste método, criamos duas variáveis, que correspondem às posições das células. Visto que já foi criado um estado, para distinguir os diferentes tipos de ambientes, apenas usamos uma condição, para verificar onde é que, no nosso terreno, existe terreno “Rural”. Após a condição ser concretizada, vamos buscar o valor dessa célula, e aplicamos o estado “Market”. O aparecimento dos mercados, no mapa, é aleatório dentro deste estado.

Na classe “Terrain”, decidimos criar também experiência com o utilizador, e para isso criamos o método *clickMarket()*. O que este método faz é, quando surge um mercado no estado “Market”, numa célula aleatória, o utilizador pode clicar em todos os mercados que apareçam no ambiente. Ao clicar, o mercado passa a estar no estado “sold”, ou seja, o utilizador vendeu o mercado, e ao ser vendido, fez com que os bens que tinha, foram adicionados aos recursos de pele e carne. Ao ser atingido o limite de recursos (pele com valor de 300 e carne de 150), adicionamos mais um mercado, para ser criado. Em código, criamos uma condição para verificar as coordenadas do rato, e de seguida, iremos buscar o valor da respetiva célula. Caso os valores do rato e da célula coincidam, é registado o clique do rato, no respetivo mercado que o utilizador selecionou.

Diagrama UML

Foi também criado um diagrama UML, onde pode ser visualizado todas as classes e métodos que foram realizadas para este projeto:

Diagrama UML dos agentes autónomos:

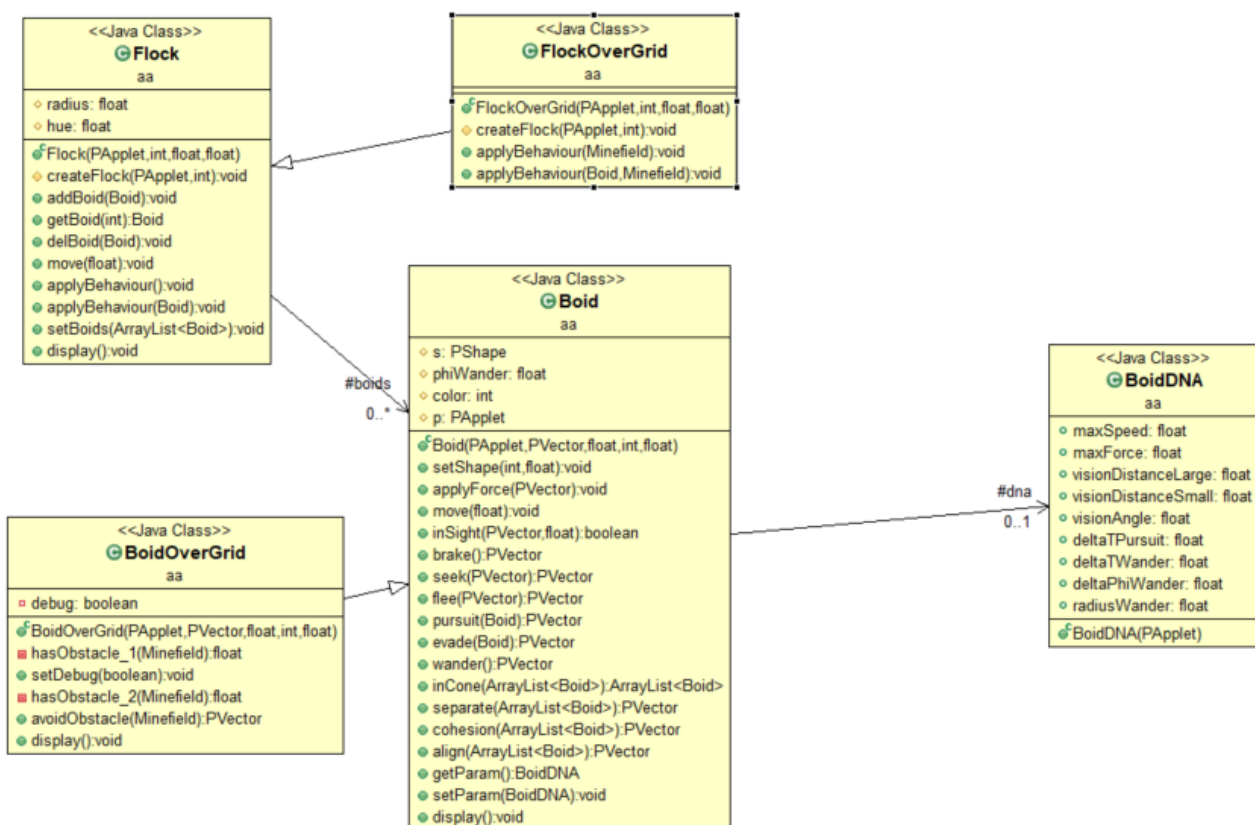


Figura 2 - diagrama UML do agente autónomo

Diagrama UML dos autómatos celulares:

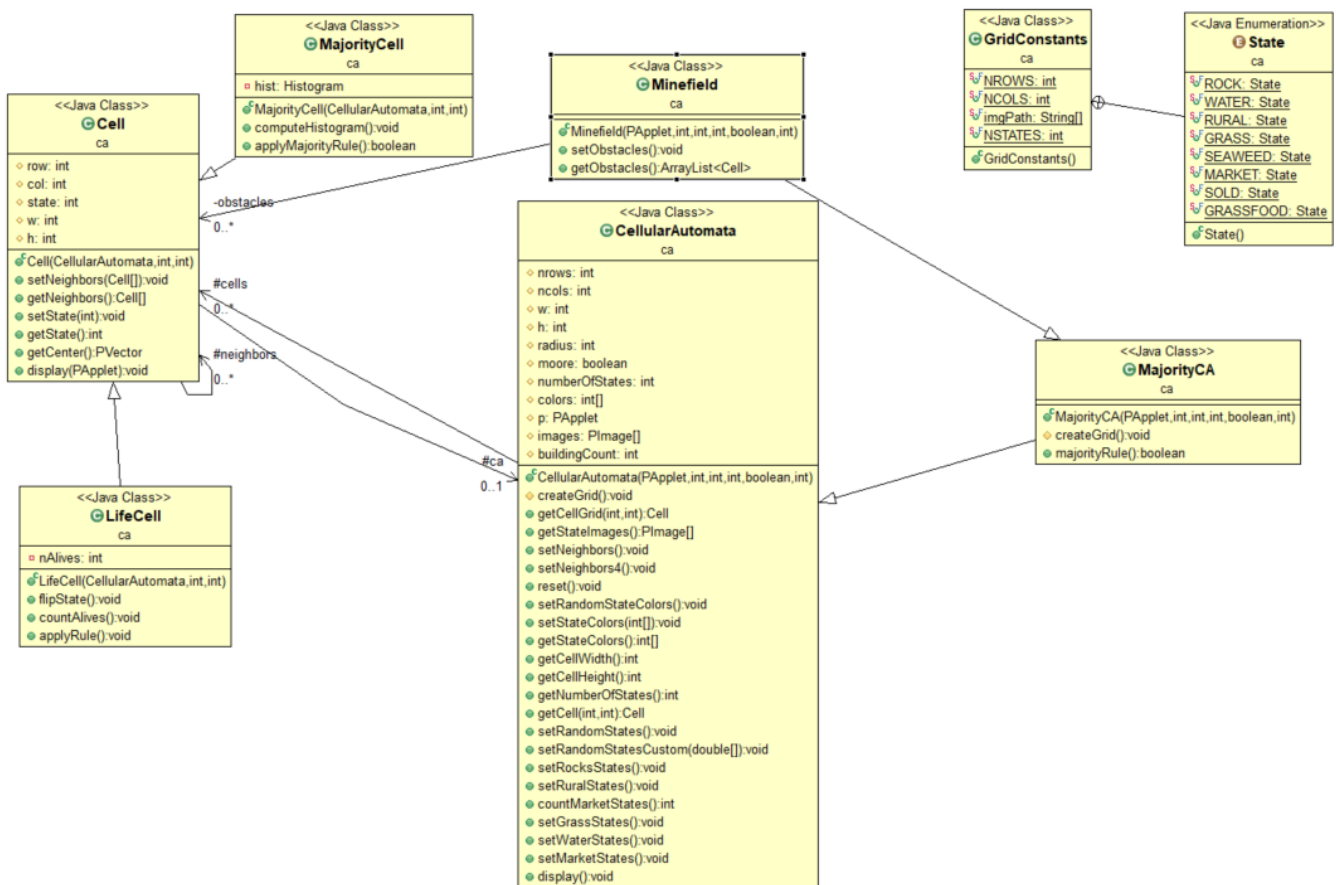


Figura 3 - diagrama UML do autómato celular

Diagrama UML do ecossistema:

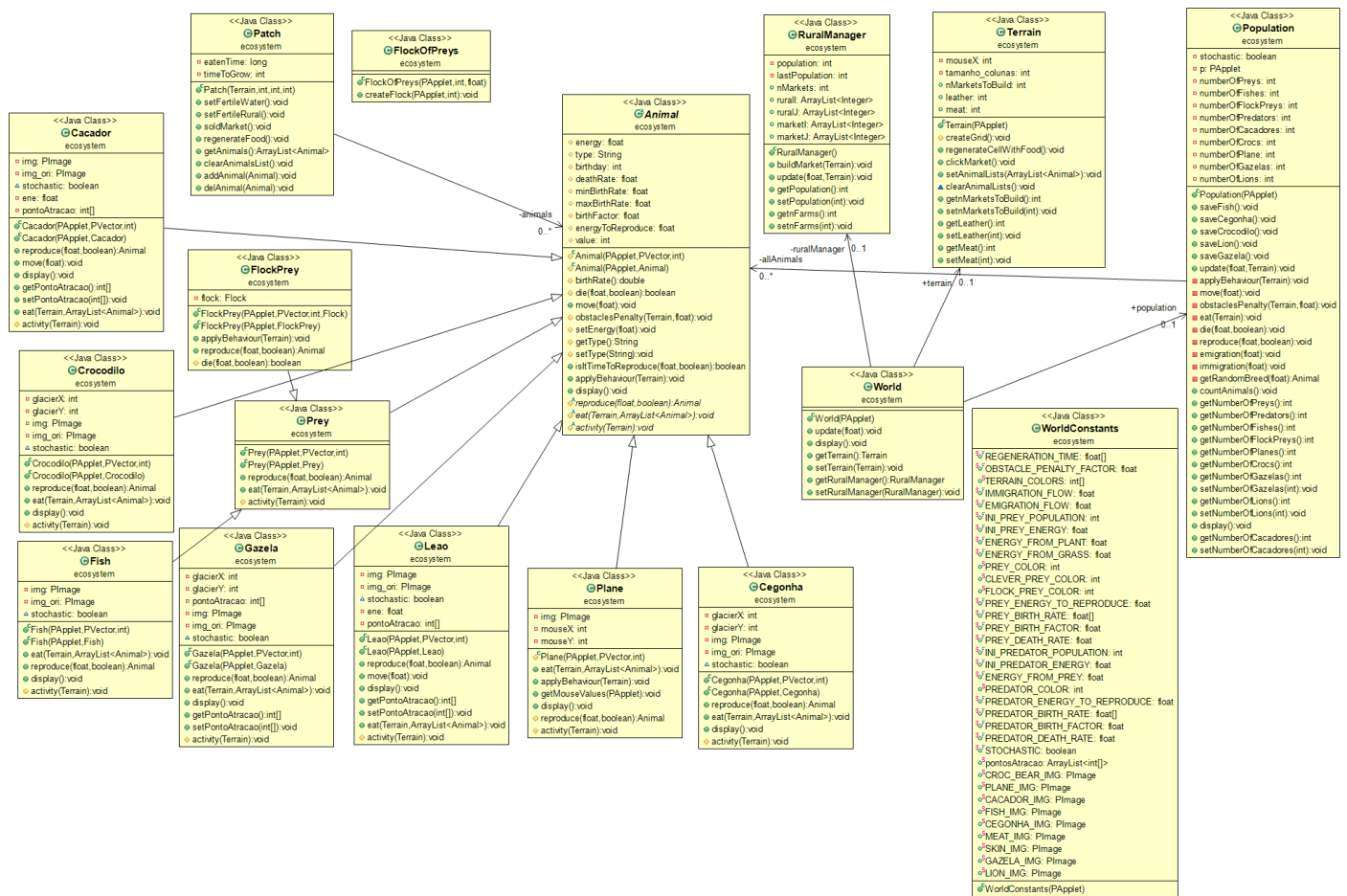


Figura 4 - diagrama UML do ecossistema

Diagrama UML dos Menus:

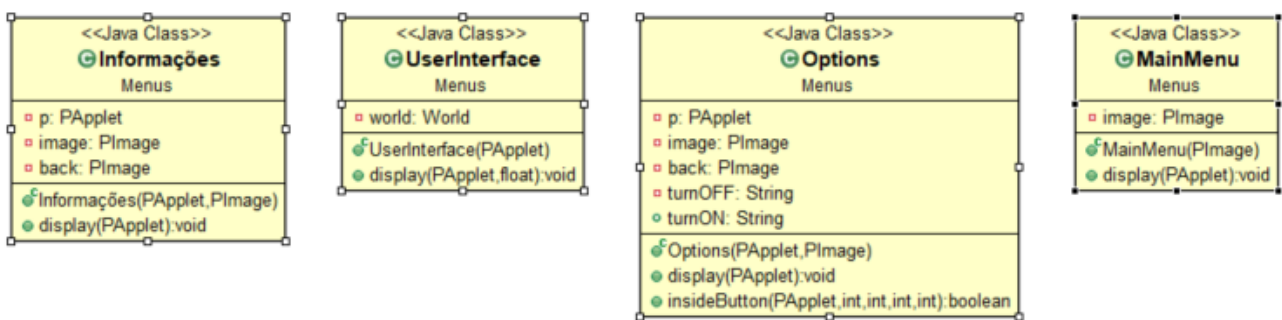


Figura 5 - diagrama UML do menu

Resultados

Os resultados verificados são os seguintes:

No ambiente aquático, o crocodilo provou-se um predador demasiado eficiente e a população de cegonhas diminuiu drasticamente ao longo do tempo. Durante algum tempo, o habitat vai ter muito menos pinguins que no início, se esperarmos um intervalo de tempo significativo observamos então a possível extinção dos pinguins e por consequência do crocodilo.



Figura 6 - resultado do ambiente aquático desenvolvido

No ambiente rural, sem a interação do utilizador conseguimos manter o número de mercados, no que toca aos caçadores conseguimos manter o número de humanos desde que estes não comecem a entrar a um ritmo elevado em ambientes prejudiciais neste caso a água.



Figura 7 - resultado do ambiente rural, com intervenção do caçador

Sem a intervenção do utilizador (click do rato nos mercados para os vender) apenas vamos observar o crescimento da quantidade de árvores. Se o utilizador intervir substancialmente, podemos observar uma grande quantidade de mercados vendidos. No entanto estas podem ser comprados por outra pessoa, tornando esse espaço uma enorme fonte de comercio de carne e pele de animais.

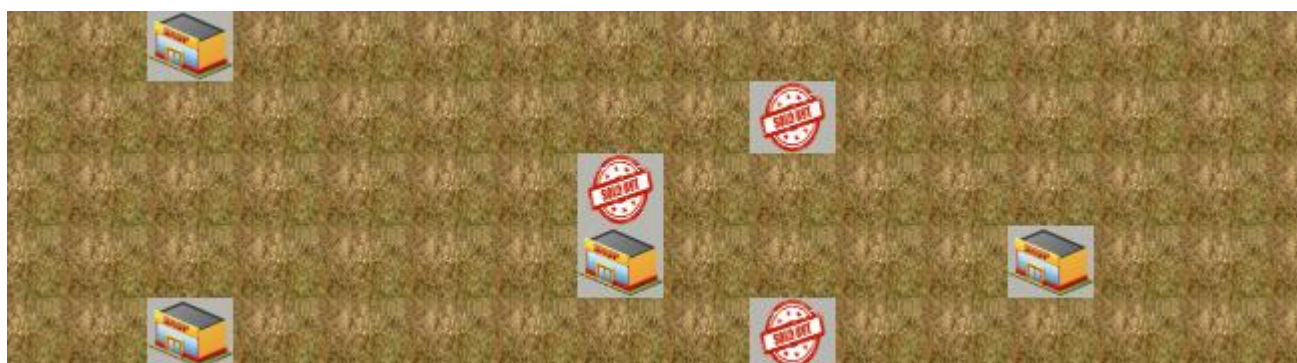


Figura 8 - resultado do comércio de carne e pele

Ao nível de experiência ao utilizador, o grupo tentou ser o mais breve e informador possível. Nas 3 figuras abaixo, podemos verificar a informação necessária para o utilizador. Na figura da esquerda, podemos verificar todas as informações necessárias de todas as espécies de presas e predadores que estão vivos no ecossistema. Na figura do meio, podemos ver os recursos que são obtidos quando selecionamos um dos mercados, aumentando o número de carne e de pele. Já na figura da direita, demos a possibilidade de ao utilizador salvar as espécies, que entrarem em vias de extinção.

User Interface








| Tipo | Quantidade |
|---|------------|
|  | 6 |
|  | 1 |
|  | 3 |
|  | 0 |
|  | 13 |
|  | 5 |
|  | 5 |

Figura 11 - informação acerca do número de espécies vivas

Recursos



| | |
|---|---|
|  | 0 |
|  | 0 |

Figura 9 - número de recursos obtidos



Figura 10 - salvar espécies

Conclusões

Com este trabalho encerra-se a componente prática da unidade curricular de MSSN tendo se cumpridos todos os objetivos do trabalho. Foi desenvolvido um jogo que requer a interação do utilizador onde se simula um ecossistema com diversos ambientes e espécies.

As maiores dificuldades no trabalho foi na otimização devido ao elevado número de comportamentos e atividades, desenvolver a interação em criar mercados e vendê-los, criar os pontos de atracção para espécies e descobrir uma maneira para as gazelas não vaguearem pelo mapa.

Bibliografia

Folhas fornecidos pelo docente Arnaldo Abrantes

<https://noctula.pt/o-que-e-sustentabilidade/>

<https://pt.slideshare.net/inessalgado/gesto-sustentvel-dos-recursos-12927409>

<http://paletton.com/#uid=1000u0klIIaFw0g0qFqFg0w0aF>