SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGII
Odbor: Informačné systémy, Softvérové inžinierstvo

HERBAL

Dokumentácia k produktu
verzia 2.98
13. 12. 2010

Vedúci projektu	Ing. Pavel Bartoš
Členovia tímu č. 19	Bc. Miloš Auder
	Bc. Andrej Belica
	Bc. Lukáš Ďurčák
	Bc. Miroslav Mikuláš
	Bc. Martin Paššák
	Bc. Ján Romaňák

Obsah

0 Úvod 0-1

0.1 Účel dokumentu 0-1

0.2 Slovník používaných pojmov 0-1

0.3 História verzií dokumentu 0-2

0.3.1 História kapitoly úvod 0-3

0.3.2 História kapitoly analýza 0-3

0.3.3 História kapitoly špecifikácia požiadaviek 0-4

0.3.4 História kapitoly návrh 0-5

0.3.5 História kapitoly návrh testovania 0-6

0.3.6 História kapitoly návrh cieľov pre finálny produkt 0-6

1 Analýza 1-1

2 Špecifikácia požiadaviek 2-1

2.1 Funkcionálne požiadavky 2-2

2.1.1 UC Spusti simuláciu 2-4

2.1.2 UC Nastav parametre sveta 2-5

2.1.3 UC Nastav druhy 2-6

2.1.4 UC Nastav parametre akcií 2-8

2.1.5 UC Načítaj nastavenia akcií zo súboru 2-8

2.1.6 UC Ulož nastavenia akcií do súboru 2-9

2.1.7 UC Nastav parametre reportovania 2-10

2.1.8 UC Zapíš štatistické údaje do súboru 2-10

2.1.9 UC Zastav simuláciu 2-11

2.1.10 UC Zapni vizualizáciu 2-11

2.1.11 UC Vypni vizualizáciu 2-12

2.1.12 UC Spusti kreslenie grafu 2-12

2.1.13 UC Označ jedinca 2-13

2.1.14 UC Ulož jedinca 2-14
2.1.15 UC Vlož jedinca 2-14
2.1.16 UC Ulož populáciu 2-16
2.1.17 UC Vlož druh 2-16
2.1.18 UC Ulož aktuálny stav simulácie 2-17
2.1.19 UC Vlož objekt 2-18
2.1.20 UC Odstráň objekt 2-19
2.1.21 UC Vytvor mapu 2-19
2.2 Nefunkcionálne požiadavky 2-20
3 Návrh 3-1
3.1 Návrh riadenia a modelu systému 3-1
3.1.1 Architektúra systému 3-1
3.1.2 Model tried 3-2
3.1.3 Návrh konkrétnych podmienok a akcií 3-9
3.1.4 Návrh ďalších aspektov simulácie 3-10
3.1.5 Návrh reportéra 3-12
3.2 Návrh vizualizácie 3-13
3.2.1 Model tried 3-13
3.2.2 GUI 3-14
3.2.3 Vizualizácia sveta 3-16
4 Návrh testovania 4-1
4.1 Hypotézy 4-1
5 Návrh cieľov pre finálny produkt 5-1

0 Úvod

0.1 Účel dokumentu

Tento dokument pozostáva z dvoch častí. Prvá časť obsahuje analýzu, špecifikáciu, návrh a opis implementácie programu vyvíjaného v rámci projektu HERBAL. Druhá časť predstavuje dokumentáciu riadenia tímu počas projektu. Projekt je vypracovaný v rámci predmetu Tímový projekt na Fakulte Informatiky a Informačných Technológií Slovenskej Technickej Univerzity v Bratislave. Rolu zákazníka/vedúceho projektu zastáva Ing. Pavel Bartos. Tím pozostáva z vedúceho tímu Bc. Martina Paššáka a ďalších členov tímu Bc. Miloša Audera, Bc. Andreja Belicu, Bc. Lukáša Ďurčáka, Bc. Miroslava Mikuláša a Bc. Jána Romaňáka.

0.2 Slovník používaných pojmov

Herb:

Pojem herb označuje akýkoľvek organizmus v programe, ktorý obsahuje genetický kód a v každom kroku simulácie vykonáva na základe tohto kódu isté činnosti.

Potrava:

Pojem potrava označuje základný zdroj energie pre herbov. Potrava môže ale nemusí obsahovať vlastný genetický kód a vykonávať akcie počas behu simulácie.

Tik/kolo

Obidva pojmy sú v tomto dokumente synonymami. Vyjadrujú vykonanie jedného tiku/kola v simulácií – počas tiku/kola vykonajú svoje akcie všetky organizmy, ktoré práve vo svete existujú.

UC

Use case – prípad použitia.

Report/reportovanie

Označuje záznam alebo proces zaznamenávania štatistických informácií o simulácií. Ide napríklad o vytváranie záznamu obsahujúceho informácie o veľkosti populácií organizmov a podobne.

0.3 História verzií dokumentu

Tab. 1 – Tabuľka verzií dokumentu

verzia	dátum	opis
1.0	2. 11. 2010	spojenie analýzy a špecifikácie požiadaviek do jedného dokumentu
1.05	3. 11. 2010	spojenie s dokumentáciou k riadeniu projektu
1.1	3. 11. 2010	doplnenie kapitoly metodiky v časti o riadení projektu
1.5	3. 11. 2010	doplnenie kapitoly o návrhu
1.6	3. 11. 2010	doplnenie úvodu zjednotenie formátovania dokumentu
1.7	4. 11. 2010	doplnenie plánu do časti o riadení projektu upravené referencie obrázkov
1.8	4. 11. 2010	skontrolovaná verzia – malé úpravy (preklepy a pod.)
1.9	4. 11. 2010	ďalšie opravy preklepov
2.0	24. 11. 2010	rozdelenie dokumentu – časť o riadení projektu je odteraz samostatný dokument
2.1	24. 11. 2010	úpravy kapitoly návrh
2.2	27. 11. 2010	úpravy všetkých kapitol
2.3	27. 11. 2010	úpravy kapitoly špecifikácia požiadaviek
2.4	30. 11. 2010	úprava kapitoly špecifikácia požiadaviek
2.5	1. 12. 2010	úprava kapitoly návrh
2.6	1. 12. 2010	úprava kapitol návrh, špecifikácia požiadaviek a analýza
2.65	2. 12. 2010	úprava kapitoly návrh
2.7	8. 12. 2010	úprava kapitoly špecifikácia požiadaviek
2.8	8. 12. 2010	úprava kapitoly návrh
2.9	8. 12. 2010	úprava kapitol špecifikácia požiadaviek a návrh
2.95	8. 12. 2010	úprava kapitoly návrh
2.96	8. 12. 2010	úpravy kapitoly špecifikácia požiadaviek

2.97	9. 12. 2010	zjednotenie nekorektného označenia verzií jednotlivých častí dokumentu oprava preklepov úpravy kapitol návrh a analýza vytvorenie kapitoly návrh testovania spolu so záznamom jej histórií
2.98	13. 12. 2010	vytvorenie kapitoly návrh cieľov pre finálny produkt

0.3.1 História kapitoly úvod

V tejto kapitole uvádzame zmeny kapitoly 0 „Úvod“ okrem zmien spočívajúcich v úpravách tabuliek verzií dokumentu a kapitol.

Tab. 2 – Tabuľka verzií kapitoly 0 „Úvod”

verzia	dátum	opis
1.0	3. 11. 2010	základ kapitoly účel dokumentu a slovník pojmov (herb a potrava)
1.1	24. 11. 2010	doplnenie slovníka pojmov (tik/kolo)
1.2	27. 11. 2010	doplnenie slovníka pojmov (UC) doplnenie kapitoly o verziách kapitol a dokumentu
1.3	30. 11. 2010	doplnenie slovníka pojmov (Report/reportovanie)

0.3.2 História kapitoly analýza

Tab. 3 – Tabuľka verzií kapitoly 1 „Analýza”

verzia	dátum	opis
1.0	16. 10. 2010	základ kapitoly informácie o umelom živote a pravidlových systémoch
1.1	20. 10. 2010	doplnenie časti konkrétne súvisiacej s projektom
1.2	27. 10. 2010	doplnenie úvodu o umelej inteligencii doplnenie časti o hypotézach
1.21	27. 10. 2010	doplnenie všetkých hypotéz
2.0	27. 11. 2010	odstránenie úvahy o moduloch/triedach, ktoré bude nutné vytvoriť – presunuté do návrhu

2.1	1. 12. 2010	preformulovanie hypotéz
2.2	9. 12. 2010	presunutie hypotéz z tejto kapitoly do novej kapitoly návrh testovania

0.3.3 História kapitoly špecifikácia požiadaviek

Tab. 4 – Tabuľka verzií kapitoly 2 „Špecifikácia požiadaviek"

verzia	dátum	opis
0.67	16. 10. 2010	funkcionálne požiadavky, svet, herbovia
0.9	16. 10. 2010	doplnenie časti o génoch a potrave, požiadavkách na simuláciu a možnosti budúcich vylepšení
1.0	20. 10. 2010	dokončená špecifikácia
1.1	25. 10. 2010	upravenie podľa pripomienok vedúceho projektu
2.0	27. 11. 2010	prepracovaná verzia označené časti na odstránenie (funkcionálne požiadavky) doplnenie prípadov použitia spusti simuláciu, nastav parametre sveta a nastav druhy doplnenie opisu stavov simulácie a vizualizácie
2.1	27. 11. 2010	doplnený prípad použitia zastav simuláciu oprava preklepov
2.2	30. 11. 2010	doplnenie viacerých prípadov použitia
2.3	30.11.2010	preformulovane nefunkcionálnych požiadaviek
2.4	1. 12. 2010	pridanie diagramu prípadov použitia a stavových diagramov
2.5	8. 12. 2010	doplnenie informácie o rozdelení prípadov na tie týkajúce sa prototypu a tie týkajúce sa ďalších verzií doplnenie niektorých prípadov použitia
2.6	8. 12. 2010	doplnenie všetkých chýbajúcich doteraz identifikovaných prípadov použitia nová verzia diagramu prípadov použitia
2.7	8. 12. 2010	doplnenie nefunkcionálnych požiadaviek

0.3.4 História kapitoly návrh

Tab. 5 – Tabuľka verzií kapitoly 3 „Návrh"

verzia	dátum	opis
1.0	3. 11. 2010	vytvorenie kapitoly návrh jadra – diagram a základný popis tried, prípady použitia a sekvenčné diagramy návrh vizualizácie – diagramy a popisy tried
2.0	24. 11. 2010	základ prepracovanej verzie doplnenie kompletného popisu tried jadra
2.1	27. 11. 2010	prepracovaná verzia označené časti na odstránenie (prípady použitia, sekvenčné diagramy) doplnenie zdôvodnenia voľby C# a XNA do úvodu návrhu doplnenie základnej dekompozície na moduly do časti architektúre (ktorá bola týmto pridaná)
2.1.1	27. 11. 2010	oprava nekorektného číslovania kapitoly (pri číslach strán nebolo uvedené číslo kapitoly)
2.2	1. 12. 2010	doplnenie podkapitoly návrh základných podmienok a akcií
2.3	1. 12. 2010	pridanie diagramov tried a diagramu balíkov
2.4	2. 12. 2010	pridanie opisu tried vizualizačnej časti
2.5	8. 12. 2010	pridanie opisu ďalších aspektov simulácie pridanie diagramu komponentov a nového diagramu balíkov
2.6	8. 12. 2010	pridanie kapitoly o návrhu reportovania
2.7	8. 12. 2010	prerobenie kapitoly o návrhu vizualizácie
2.8	9. 12. 2010	úpravy formátovania opravy preklepov

0.3.5 História kapitoly návrh testovania

Tab. 6 – Tabuľka verzií kapitoly „Návrh testovania"

verzia	dátum	opis
1.0	9. 12. 2010	vytvorenie kapitoly presunutie hypotéz z analýzy do tejto kapitoly

0.3.6 História kapitoly návrh cieľov pre finálny produkt

Tab. 7 – Tabuľka verzií kapitoly „Návrh cieľov pre finálny produkt"

verzia	dátum	opis
1.0	13. 12. 2010	vytvorenie kapitoly

1 Analýza

Umelá inteligencia sa ako vedný odbor objavila v polovici 50.-tych rokov minulého storočia – samotný pojem sa prvýkrát objavil na konferencii v Dartmouth College v júli 1956 [1]. Odvtedy boli hlavnými záujmami umelej inteligencie skúmanie a návrh inteligentných agentov – agentov, ktorí by boli schopní vnímať svoje okolie a vykonávať rôzne kroky za účelom toho, aby maximalizovali efektívnosť svojej práce. V oblasti sa vytvorili dva prístupy k vytvoreniu inteligencie – prístup zhora-nadol a prístup zdola-nahor. Samotná oblasť sa takisto rozdelila na veľké množstvo „smerov". S ich postupným rozvojom sa vedci pokúšali o čoraz komplexnejšie simulácie a po čase narazili na hardvérovú prekážku v podobe fyzických robotov reprezentujúcich agenty. Aby sa táto prekážka obišla, začalo sa pristupovať k simulácií agentov v počítačových systémoch. Tak sa v 80.-tych rokoch objavila aj oblasť umelého života.

Umelý život (ang.: Artificial Life, Alife) je „oblasť výskumu zameraná na porozumenie života prostredníctvom pokusov o abstrakciu základných dynamických princípov za týmto biologickým fenoménom a na rekonštrukciu tohto dynamického systému v inom médiu tak, aby ho bolo možné študovať a manipulovať s ním novými spôsobmi"[2]. Táto oblasť bola takto pomenovaná Christopherom Langtonom v roku 1986. Umelý život sa delí na tri podskupiny – soft Alife (založený na softvérovej reprezentácií), hard Alife (hardvérová reprezentácia) a wet Alife (biochemicky vytvorený umelecký život). Pojem umelý život sa často používa aj na označenie samostatnej prvej skupiny – softvérovo reprezentovaného umelého života. Práve tento druh je predmetom tohto projektu. Umelý život sa od mnohých oblastí umelej inteligencie odlišuje tým, že používa prístup zdola-nahor – inteligentné správanie sa snaží dosiahnuť prostredníctvom spolupráce jednoduchých základných zložiek.

V softvérovom ponímaní je možné umelý život naprogramovať ako sekvenčný evolučný algoritmus využívajúci istú sadu inštrukcií. V tomto projekte sa však budeme zaoberať alternatívnou reprezentáciou – pravidlovým systémom. Gény a teda aj samotné správanie organizmov bude charakterizované systémom pravidiel v tvare *„Podmienka → Akcia"*. Takýto pravidlový systém bude pozostávať z množiny pravidiel, ktoré sa budú skladať zo základných podmienok a akcií. Tieto podmienky a akcie budú do pravidiel komponované s úmyslom vytvoriť jedinečné vzory správania pre konkrétne implementované druhy organizmov a ich kompozícia bude teda špecifikovať jednotlivé druhy. Množiny základných podmienok a akcií budú vytvorené na základe špecifikácie požiadaviek pre projekt.

Pravidlový systém vo všeobecnosti okrem množiny pravidiel vyžaduje aj existenciu mechanizmu, ktorý rozhoduje o tom, ktoré konkrétne pravidlá z množiny znalostí sa využijú v jednotlivých krokoch spracovávania vstupov. Existujú dva základné druhy mechanizmov – sekvenčný a náhodný. Sekvenčný mechanizmus prechádza pravidlá sekvenčne a na aplikáciu vyberie prvé pravidlo, ktoré spĺňa podmienku. Náhodný mechanizmus, na rozdiel od sekvenčného, prechádza pravidlá v náhodnom poradí. Kombináciou týchto dvoch prístupov je možné vytvoriť ďalšie druhy mechanizmov.

V rámci tohto projektu bude nutné vytvoriť model sveta, v ktorom sa bude umelý život rozvíjať. Tento model je možné charakterizovať ako systém, v ktorom bude interagovať veľké množstvo

rôznych entít (jednotlivé organizmy, svet, potrava, zásahy od používateľa). Preto je vhodné tento model reprezentovať objektovo orientovaným prístupom.

Jednotlivé skupiny entít (napríklad: model sveta, kontrolný mechanizmus, logovanie, ...) bude vhodné implementovať v rámci samostatných modulov, ktoré budú len minimálne previazané. Takéto riešenie je vhodné, lebo umožňuje jednoduché pridávanie alebo zmenu individuálnych komponentov, ktoré neovplyvní činnosť celého systému. S týmto riešením je takisto spojená nutnosť vytvoriť kvalitné rozhrania medzi modulmi tak, aby bolo mužné ľahko prenášať medzi modulmi potrebné dáta.

Pri implementácií bude nutné klásť dôraz aj na efektivitu vykonávania aplikácie. Hlavným jej účelom totiž bude vykonávanie simulácie, z ktorých sa budú vyvodzovať rôzne závery. Aby bolo možné vyvodiť tieto závery kvalitne, bude potrebné pracovať s čo možno najväčšími populáciami a veľkosťou sveta a tieto môžu mať značný negatívny vplyv na rýchlosť vykonávania programu. Náhodný charakter simulácií takisto vyžaduje ich mnohonásobné vykonávanie aby bolo možné získané dáta považovať za dostatočne reprezentatívne – toto je ďalší dôvod zvýrazňujúci dôležitosť rýchlosti vykonávania simulácií.

Zdroje:

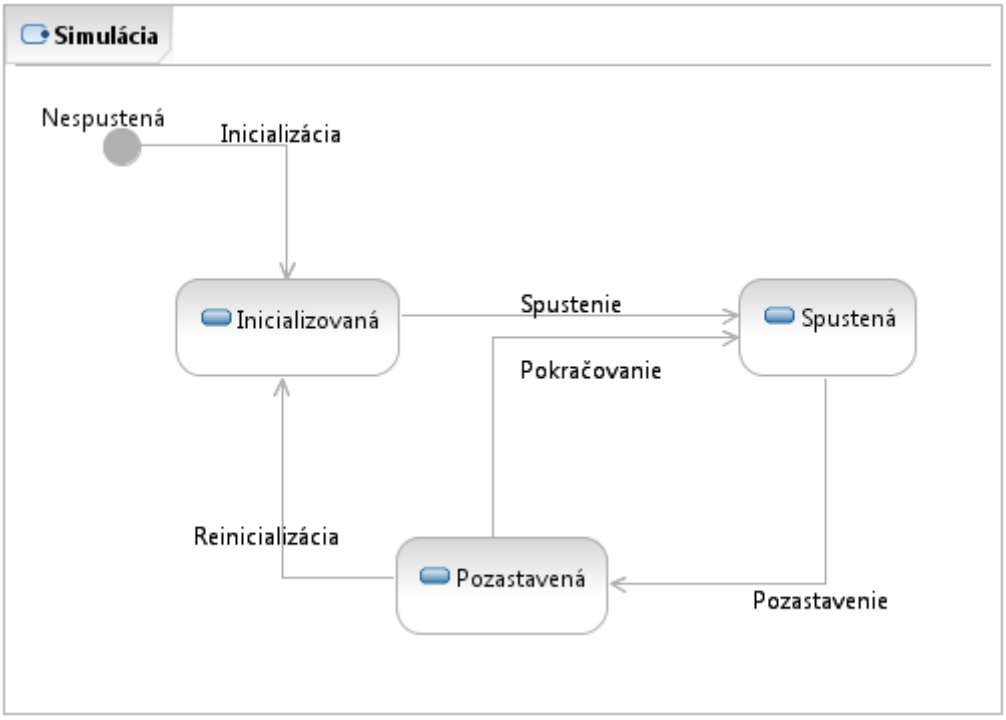
[1]Brunette, E.S.; Flemmer, R.C.; Flemmer, C.L.: A review of artificial intelligence, *4th International Conference on Autonomous Robots and Agents 2009*, pp.385-392, 2009

[2]Langton, C. G.: Preface In *Artificial Life II, SFI Studies in the Sciences of Complexity*, vol. 10, Addison-Wesley, 1992.

[3]Bartoš, P.: Pracovný materiál číslo 2, 2010.

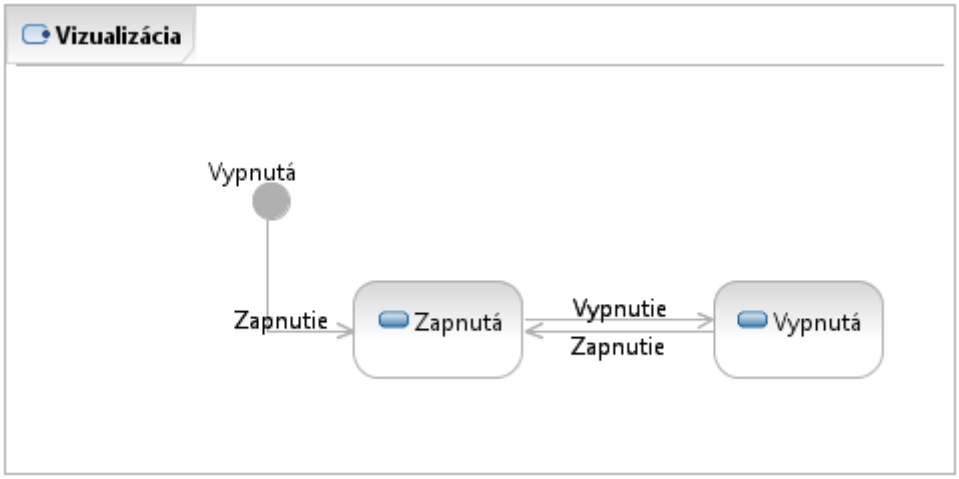
2 Špecifikácia požiadaviek

Vychádzajúc zo zadania, pracovného textu ktorý sme obdržali od zákazníka a z diskusií, sme identifikovali požiadavky uvedené v kapitolách 2.1 „Funkcionálne požiadavky“ a 2.2 „Nefunkcionálne požiadavky“. Pri opisoch jednotlivých požiadaviek sa vyskytujú referencie na simuláciu, vizualizáciu a ich stavy počas behu aplikácie. Preto na obrázkoch (Obr. 1 a Obr. 2) uvádzame stavové diagramy simulácie a vizualizácie spolu s opismi stavov.



Obr. 1 - Stavový diagram simulácie

Simulácia začína v stave „nespustená“ – v tomto stave simulácia neprebieha. Zo stavu „nespustená“ je možný prechod do stavu „inicializovaná“ – v tomto stave sú pre simuláciu nastavené parametre a je vytvorený model sveta, ktorý bude simulovaný. Zo stavu „inicializovaná“ je možný prechod do stavu „spustená“ – v tomto stave simulácia prebieha a v modeli plynú tiky simulácie. Zo stavu „spustená“ je možný prechod do stavu „pozastavená“ – v tomto stave simulácie neprebieha – došlo k jej zastaveniu na konci posledného vykonávaného tiku (aplikácia vždy dokončí momentálne vykonávaný tik a až potom simuláciu zastaví). Zo stavu „pozastavená“ sú možné dva prechody – do stavu „spustená“ – simulácia pokračuje nasledujúcimi tikmi, alebo do stavu „inicializovaná“ – pri tomto prechode sa model sveta nanovo inicializuje nastavenými parametrami a doterajší priebeh simulácie bude odstránení z pamäte aplikácie.



Obr. 2 - Stavový diagram vizualizácie

Vizualizácia začína v stave „vypnutá“ – v tomto stave vizualizácia neprebieha. Zo stavu „vypnutá“ je možný prechod do stavu „zapnutá“ – tento prechod vytvorí grafické okno vizualizácie a zobrazuje v ňom momentálny model sveta. Požiadavkou na tento prechod je to, aby simulácie bola v jednom zo stavov: „inicializovaná“, „spustená“ alebo „pozastavená“. Zo stavu „zapnutá“ je možný prechod do stavu „vypnutá“ – tento prechod vypne vizualizáciu a zatvorí jej okno.

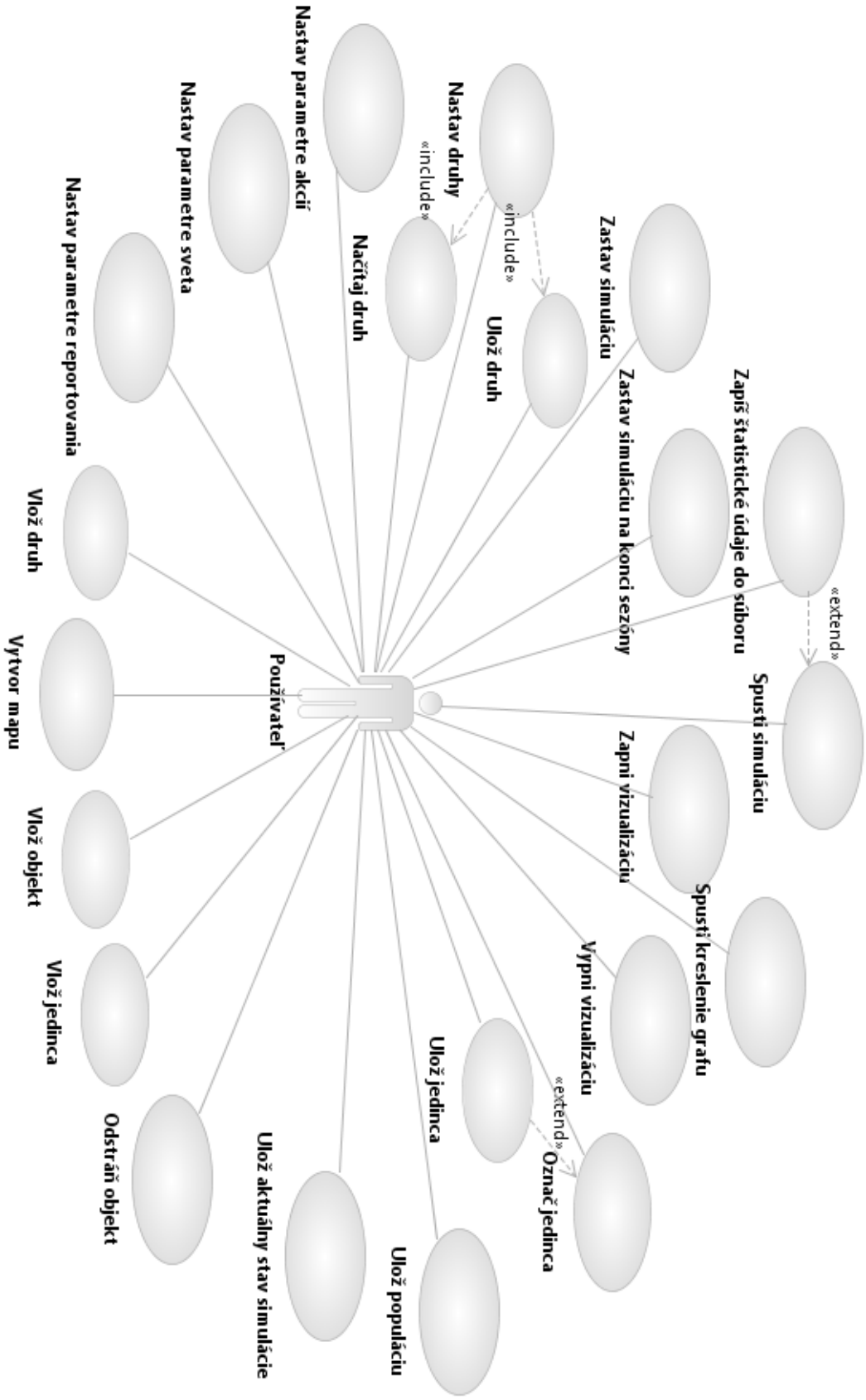
2.1 Funkcionálne požiadavky

V tejto časti dokumentácie uvedieme základné funkcionálne požiadavky, ktoré sme po konzultáciách so zákazníkom/vedúcim projektu identifikovali. Túto časť dokumentácie vypracoval Ján Romaňák.

S ohľadom na to, že aplikácia bude slúžiť najmä na vykonávanie veľkého množstva simulácií s rôznymi parametrami používateľom, sme identifikovali prípady použitia, ktoré sú uvedené na obrázku (Obr. 3).

Jednotlivé prípady použitia sú uvedené a bližšie opísané v nasledujúcich podkapitolách. Názvy týchto kapitol z dôvodu zvýšenia prehľadnosti začínajú skratkou „UC“ (angl.: Use Case). Pri opise možností akcií v grafickom rozhraní sú v prípadoch použitia uvádzané aj návrhy názvov ovládacích prvkov, ktoré budú danú akciu vykonávať (napríklad: zvolí možnosť inicializácie sveta („Init“)).

Podkapitoly 2.1.1 až 2.1.11 opisujú prípady použitia, ktoré sa vzťahujú k prototypu aplikácie. Ostatné podkapitoly (2.1.12 až 2.1.21) opisujú dodatočné prípady použitia finálneho produktu.



Obr. 3 - Diagram prípadov použitia

2.1.1 UC Spusti simuláciu

Prípad použitia „Spusti simuláciu“ je základným prípadom použitia aplikácie, keďže napĺňa hlavný účel aplikácie.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu

Výstupné podmienky

V aplikácii je spustená simulácia

Hlavný tok

1. používateľ v grafickom rozhraní zvolí možnosť inicializácie sveta („Init“)
2. aplikácia vytvorí model sveta, druhy a ich počiatočné populácie podľa základných, v aplikácií prednastavených parametrov
3. používateľ v grafickom rozhraní zvolí možnosť spustenia simulácie („Start“)

Body rozšírenia

- Vykonávanie simulácie – po vykonaní kroku 3 hlavného toku

Alternatívny tok 1 – UC Nastav parametre sveta

Aktivuje sa namiesto kroku 1 hlavného toku. Spočíva vo vykonaní prípadu použitia „Nastav parametre sveta“ (bližšie opísaný v kapitole 2.1.2 „UC Nastav parametre sveta“). Po jeho vykonaní sa pokračuje krokom 3 hlavného toku.

Alternatívny tok 2 – UC Nastav druhy

Aktivuje sa namiesto kroku 1 hlavného toku. Spočíva vo vykonaní prípadu použitia „Nastav druhy“ (bližšie opísaný v kapitole 2.1.3 „UC Nastav druhy“). Po jeho vykonaní sa pokračuje krokom 3 hlavného toku.

Alternatívny tok 3 – Načítanie uloženej simulácie

Aktivuje sa namiesto kroku 1 hlavného toku. Umožňuje používateľovi inicializovať svet pomocou súboru s uloženým stavom simulácie, ktorá bola vykonávaná v minulosti. Po jeho vykonaní sa pokračuje krokom 3 hlavného toku. Požiadavka na tento alternatívny tok sa nevzťahuje k prototypu produktu, ale až k jeho finálnej verzii.

1. používateľ zvolí možnosť načítania uloženého stavu simulácie

2. aplikácia zobrazí okno prehliadača súborov
3. používateľ v otvorenom okne prehliadača súborov vyberie súbor s uloženým stavom simulácie a potvrdí svoj výber („OK“)
4. aplikácia skontroluje, či používateľ zvolil validný súbor stavu simulácie – ak je súbor validný, pokračuje sa krokom 5 tohto toku, v opačnom prípade sa pokračuje krokom 3 tohto toku
5. aplikácia zatvorí okno prehliadača súborov a vytvorí model sveta podľa rekonštrukcie uloženého stavu zo súboru

2.1.2 UC Nastav parametre sveta

Tento prípad použitia umožňuje používateľovi nastaviť pre simuláciu ľubovoľné parametre (samozrejme v rámci povolených parametrov – aplikácia bude kontrolovať ich správnosť (dátový typ a podobne)).

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácie nie je v stave spustená

Výstupné podmienky

V aplikácii sú pre simuláciu nastavené používateľom zvolené parametre

Hlavný tok

1. používateľ zvolí v grafickom rozhraní možnosť nastavenia parametrov sveta („World settings“)
2. aplikácia zobrazí okno umožňujúce nastaviť parametre sveta
3. používateľ zadá ním zvolené parametre do polí na to určených v otvorenom okne nastavení parametrov sveta
4. aplikácia skontroluje korektnosť zvolených hodnôt parametrov; ak sú korektne zadané, pokračuje sa krokom 5, v opačnom prípade sa pokračuje znovu krokom 3
5. používateľ zvolí možnosť aplikovania zvolených nastavení parametrov („OK“)
6. aplikácia zatvorí okno nastavení parametrov sveta

Alternatívny tok 1 – Načítaj nastavení parametrov sveta zo súboru

Tento alternatívny tok umožňuje používateľovi rýchle nastavenie všetkých parametrov sveta pomocou súboru z uloženými nastaveniami. Aktivuje sa namiesto kroku 3 hlavného toku, po vykonaní tohto tou sa pokračuje krokom 5 hlavného toku.

1. používateľ zvolí možnosť načítania parametrov zo súboru („Load from file“)
2. aplikácia otvorí okno prehliadača súborov
3. používateľ v otvorenom okne prehliadača súborov nájde súbor s parametrami a zvolí ho možnosťou v grafickom rozhraní okna („OK“); používateľ má takisto možnosť okno prehliadača súborov zatvoriť – týmto skončí tento alternatívny tok a pokračuje sa krokom 3 hlavného toku
4. aplikácia skontroluje korektnosť zadaného súboru – ak je validným súborom nastavení parametrov sveta pokračuje sa krokom 5 tohto toku, ak nie je validným súborom nastavení parametrov sveta upozorní na to používateľa a pokračuje sa krokom 3 tohto toku
5. aplikácia zatvorí okno prehliadača súborov a do okna nastavení parametrov sveta zobrazí nastavenia parametrov načítané zo zadaného súboru

Alternatívny tok 2 – Uloženie nastavení parametrov sveta do súboru

Tento prípad použitia umožňuje používateľovi uložiť momentálne nastavenia parametrov sveta do súboru pre ďalšie využitie. Aktivuje sa po kroku 5 hlavného toku a po jeho vykonaní sa pokračuje krokom 6 hlavného toku.

1. používateľ zvolí možnosť uložení nastavení parametrov sveta do súboru („Save to file“)
2. aplikácia otvorí okno prehliadača súborov
3. používateľ sa v okne prehliadača súborov dostane do požadovaného adresára a zadá meno súbor, ktorý sa má vytvoriť
4. aplikácia vytvorí zadaný súbor nastavení parametrov sveta v zadanom adresári

Alternatívny tok 3 – UC Vytvor mapu

Aktivuje sa namiesto kroku 3 hlavného toku. Spočíva vo vykonaní prípadu použitia „Nastav Vytvor mapu“ (bližšie opísaný v kapitole 2.1.21 „UC Vytvor mapu“). Po jeho vykonaní sa pokračuje krokom 3 hlavného toku.

2.1.3 UC Nastav druhy

Tento prípad použitia umožňuje používateľovi nastaviť druhy organizmov, ktoré budú vo svete simulované.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácie nie je v stave spustená

Výstupné podmienky

V aplikácii sú nastavené používateľom zvolené druhy organizmov

Hlavný tok

1. používateľ v grafickom rozhraní aplikácie zvolí možnosť nastavenia druhov („Species settings“)
2. aplikácia otvorí okno nastavení druhov
3. používateľ zadá ním zvolené parametre druhu do polí na to určených v okne nastavení druhov a zvolí možnosť pridania druhu do zoznamu druhov („Add this species“)
4. aplikácia skontroluje korektnosť zvolených hodnôt parametrov; ak sú korektne zadané, pokračuje sa krokom 5, v opačnom prípade sa pokračuje znovu krokom 3
5. používateľ má možnosť vytvoriť nový druh – pokračuje sa krokom 3, alebo má možnosť aplikovať doteraz nastavené druhy voľbou v grafickom rozhraní („Apply“) – v tomto prípade sa pokračuje krokom 6
6. aplikácia zatvorí okno nastavení druhov

Alternatívny tok 1 – Načítanie druhu zo súboru

Tento tok umožňuje používateľovi načítať nastavenia druhu zo súboru. Aktivuje sa namiesto kroku 3 hlavného toku a po jeho vykonaní sa pokračuje krokom 5 hlavného toku.

1. používateľ zvolí možnosť načítania nastavení druhu zo súboru („Load from file“)
2. aplikácia otvorí okno prehliadača súborov
3. používateľ v otvorenom okne prehliadača súborov nájde súbor s nastaveniami druhu a zvolí ho možnosťou v grafickom rozhraní okna („OK“); používateľ má takisto možnosť okno prehliadača súborov zatvoriť – týmto skončí tento alternatívny tok a pokračuje sa krokom 3 hlavného toku
4. aplikácia skontroluje korektnosť zadaného súboru – ak je validným súborom nastavení druhu pokračuje sa krokom 5 tohto toku, ak nie je validným súborom nastavení druhu upozorní na to používateľa a pokračuje sa krokom 3 tohto toku
5. aplikácia zatvorí okno prehliadača súborov a do okna nastavení druhu pridá druh načítaný zo zadaného súboru

Alternatívny tok 3 – Uloženie druhu do súboru

Tento alternatívny tok umožňuje používateľovi uložiť nastavený druh do súboru pre ďalšie využitie. Aktivuje sa po kroku 3 hlavného toku a po jeho vykonaní sa pokračuje krokom 5 hlavného toku.

1. používateľ zvolí možnosť uložení nastavení druhu do súboru („Save to file“)
2. aplikácia otvorí okno prehliadača súborov

3. používateľ sa v okne prehliadača súborov dostane do požadovaného adresára a zadá meno súbor, ktorý sa má vytvoriť

4. aplikácia vytvorí zadaný súbor nastavení druhu v zadanom adresári

2.1.4 UC Nastav parametre akcií

Tento prípad použitia umožňuje používateľovi nastaviť parametre akcií, ktoré sa budú nachádzať v génoch organizmov. Medzi tieto parametre patria napríklad náročnosť akcie (koľko energie sa jej vykonaním spotrebuje) alebo dĺžka vykonávania akcie.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácie nie je v stave spustená

Výstupné podmienky

V aplikácii sú nastavené používateľom zvolené parametre akcií

Hlavný tok

1. používateľ zvolí v grafickom rozhraní možnosť nastavenia parametrov akcií („Action settings“)

2. aplikácia otvorí okno nastavenia parametrov akcií

3. používateľ zadá ním zvolené parametre akcií do polí na to určených v okne nastavení parametrov akcií

4. aplikácia skontroluje korektnosť zadaných hodnôt; ak sú korektné, pokračuje sa krokom 5, v opačnom prípade sa pokračuje krokom 3

5. používateľ zvolí možnosť aplikovania zvolených nastavení parametrov („OK“)

6. aplikácia zatvorí okno nastavení parametrov akcií

Body rozšírenia

- Načítanie základných hodnôt – krok 2 hlavného toku
- Uloženie hodnôt – krok 5 hlavného toku

2.1.5 UC Načítaj nastavenia akcií zo súboru

Tento prípad použitia je vykonávaný aplikáciou a vykonáva službu používateľovi – načítava predtým uložené (alebo základné, ak používateľ predtým parametre neuložil) parametre akcií zo súboru.

Tento prípad použitia rozširuje prípad použitia „Nastav parametre akcií“ (bližšie opísaný v kapitole 2.1.4 „UC Nastav parametre akcií“) v jeho bode rozšírenia „Načítanie základných hodnôt “.

Aktéri

Aplikácia

Vstupné podmienky

Používateľ spustil aplikáciu, v aplikácií je spustené okno nastavenia parametrov akcií

Výstupné podmienky

Nastavenia parametrov akcií sú načítané zo súboru a nastavené v aplikácií

Hlavný tok

1. aplikácia načíta zo súboru (ktorého umiestnenie bude určené v kóde aplikácie) uložené parametre akcií
2. aplikácia zobrazí načítané hodnoty parametrov akcií v okne nastavenia parametrov akcií

2.1.6 UC Ulož nastavenia akcií do súboru

Tento prípad použitia je vykonávaný aplikáciou a vykonáva službu používateľovi – ukladá nastavené parametre akcií do súboru pre ďalšie použitie. Tento prípad použitia rozširuje prípad použitia „Nastav parametre akcií“ (bližšie opísaný v kapitole 2.1.4 „UC Nastav parametre akcií“) v jeho bode rozšírenia „Uloženie hodnôt“.

Aktéri

Aplikácia

Vstupné podmienky

Používateľ spustil aplikáciu, v aplikácií je spustené okno nastavenia parametrov akcií

Výstupné podmienky

Nastavenia parametrov akcií sú zapísané v súbore

Hlavný tok

3. aplikácia získa hodnoty parametrov akcií, ktoré používateľ zadal v okne nastavení parametrov akcií
4. aplikácia uloží tieto parametre do súboru (ktorého umiestnenie bude určené v kóde aplikácie)

2.1.7 UC Nastav parametre reportovania

Tento prípad použitia umožňuje používateľovi nastaviť parametre reportovania – t.j. používateľ môže nastaviť ako často sa bude vykonávať reportovanie, aké štatistické údaje budú reportované, či sa bude robiť reportovanie do súboru, do grafického okna, ...

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácie nie je v stave spustená

Výstupné podmienky

V aplikácii sú nastavené používateľom zvolené parametre reportovania

Hlavný tok

1. používateľ zvolí v grafickom rozhraní možnosť nastavenia parametrov reportovania („Report settings“)
2. aplikácia otvorí okno nastavenia parametrov reportovania
3. používateľ zadá ním zvolené parametre reportovania do polí na to určených v okne nastavení parametrov reportovania
4. aplikácia skontroluje korektnosť zadaných hodnôt; ak sú korektné, pokračuje sa krokom 5, v opačnom prípade sa pokračuje krokom 3
5. používateľ zvolí možnosť aplikovania zvolených nastavení parametrov („OK“)
6. aplikácia zatvorí okno nastavení parametrov reportovania

2.1.8 UC Zapíš štatistické údaje do súboru

Tento prípad použitia je vykonávaný aplikáciou a vykonáva službu používateľovi – v pravidelných intervaloch vykonáva reportovanie aktuálneho stavu simulácie a tieto údaje zapisuje do súboru. Tento prípad použitia rozširuje prípad použitia „Spusti simuláciu“ (bližšie opísaný v kapitole 2.1.1 „UC Spusti simuláciu“) v jeho bode rozšírenia „Vykonávanie simulácie “.

Aktéri

Aplikácia

Vstupné podmienky

Používateľ spustil aplikáciu, v aplikácií je spustená simulácia

Výstupné podmienky

Aktuálny stav simulácie je zapísaný v súbore

Hlavný tok

1. aplikácia získa zo simulácie jej aktuálny stav, pričom zaznamená tie údaje, ktoré sa majú zaznamenávať podľa nastavení vykonaných v prípade použitia „Nastav parametre reportovania“ (bližšie opísaný v kapitole 2.1.7 „UC Nastav parametre reportovania“); ak tento prípad použitia nebol vykonaný použijú sa základné nastavenia reportovania určené v kóde aplikácie
2. aplikácia zapíše zaznamenané štatistické údaje do súboru (ktorého umiestnenie je určené v kóde aplikácie)

2.1.9 UC Zastav simuláciu

Tento prípad použitia umožňuje používateľovi pozastaviť vykonávanie simulácie.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácie je v stave spustená

Výstupné podmienky

Simulácie je v stave pozastavená

Hlavný tok

1. Používateľ v grafickom rozhraní aplikácie zvolí možnosť pozastavenia simulácie („Stop“)
2. Aplikácia dokončí vykonávanie momentálneho tiku a potom pozastaví simuláciu
3. Aplikácia aktualizuje zobrazované štatistické informácie o svete

2.1.10 UC Zapni vizualizáciu

Tento prípad použitia umožňuje používateľovi spustiť v aplikácií vizualizáciu simulácie.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, používateľ vykonal prvý krok hlavného toku „Spusti simuláciu“ (bližšie opísaný v kapitole 2.1.1 „UC Spusti simuláciu“) t.j. inicializoval svet („Init“), vizualizácia je v stave nespustená

Výstupné podmienky

Je otvorené grafické okno vizualizácie, vizualizácia je v stave spustená

Hlavný tok

1. používateľ zvolí v grafickom rozhraní možnosť spustenia vizualizácie („Run visualisation“)
2. ak v aplikácii nebolo počas behu aplikácie ešte vytvorené grafické okno vizualizácie, aplikácia okno vytvorí; v opačnom prípade sa v nasledujúcom kroku použije už existujúce okno vizualizácie
3. aplikácia zobrazí v grafickom okne vizualizácie aktuálny stav (vykreslí mapu sveta s existujúcimi organizmami, ...) sveta a pokračuje v jeho pravidelnom obnovovaní

2.1.11 UC Vypni vizualizáciu

Tento prípad použitia umožňuje používateľovi vypnúť v aplikácií vizualizáciu simulácie.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, vizualizácia je v stave spustená

Výstupné podmienky

Vizualizácia je v stave nespustená

Hlavný tok

1. používateľ v grafickom okne vizualizácie zvolí možnosť vypnutia vizualizácie („Turn off visualization“)
2. aplikácia pozastaví obnovovanie grafickej podoby sveta (vykresľovanie mapy sveta) v grafickom okne vizualizácie

2.1.12 UC Spusti kreslenie grafu

Tento prípad použitia umožňuje používateľovi spustiť vykresľovanie grafu štatistických informácií o simulácií v reálnom čase.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácie je v stave spustená

Výstupné podmienky

V aplikácii je spustené a v reálnom čase obnovované vykresľovanie grafu zvolených štatistických informácií

Hlavný tok

1. používateľ v grafickom okne vizualizácie zvolí možnosť spustenia kreslenia grafu
2. aplikácia otvorí okno grafu a začne vykresľovať graf pre základné v aplikácií prednastavené štatistické údaje

Alternatívny tok 1 – Nastavenie vykresľovania dát

Tento tok umožňuje používateľovi ovplyvniť, ktoré štatistické údaje sa budú v grafe zobrazovať. Aktivuje sa namiesto kroku 2 hlavného toku.

1. počas vykresľovania grafu používateľ v okne grafu zvolí možnosť pridania alebo odobrania štatistických dát z grafu
2. aplikácia okamžite prekreslí graf tak, aby boli do celého priebehu grafu pridané alebo z celého priebehu grafu odobrané príslušné dáta

2.1.13 UC Označ jedinca

Tento prípad použitia umožňuje používateľovi vo vizualizačnom okne označiť jeden zvolený organizmus počas behu simulácie a sledovať potom informácie o tomto organizme. Samotné označenie je možné vykonať len keď je simulácia pozastavená, avšak po označení je možné simuláciu znova spustiť a jedinec zostane označený a informácie o ňom sa budú naďalej obnovovať.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácie je v stave pozastavená

Výstupné podmienky

V aplikácií je zvolený jedinec evidovaný ako označený, je zvýraznený vo vizualizačnom okne a sú zobrazované informácie o ňom

Hlavný tok

1. používateľ kliknutím myši označí vo vizualizačnom okne zvoleného jedinca
2. aplikácia viditeľným spôsobom tohto označeného jedinca vo vizualizačnom okne zvýrazní
3. aplikácia zobrazí v informačnom panely v rámci vizualizačného okna základné informácie o označenom jedincovi – t.j. jeho vek, hladinu energie, jeho dna a podobne

Body rozšírenia

- Označenie jedinca – po vykonaní kroku 1 hlavného toku

2.1.14 UC Ulož jedinca

Tento prípad použitia umožňuje používateľovi uložiť do súboru informácie o momentálne označenom jedincovi (označenie jedinca je bližšie opísané v kapitole 2.1.13 „UC Označ jedinca“). Tento prípad použitia zároveň rozširuje prípad použitia „Označ jedinca“, ktorý je bližšie opísaný vo vyššie uvedenej kapitole, v jeho bode rozšírenia „Označenie jedinca“.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, v simulácií je označený nejaký organizmus

Výstupné podmienky

Informácie o označenom organizme sú zapísané do zvoleného súboru

Hlavný tok

1. používateľ v grafickom okne vizualizácie zvolí možnosť uloženia označeného jedinca
2. aplikácia otvorí prehliadač súborov
3. používateľ zvolí adresár a meno súboru, ktorý bude obsahovať informácie o organizme a svoju voľbu potvrdí („OK“)
4. aplikácia vytvorí nový súbor v zadanom adresári so zadaným menom a uloží do tohto súboru informácie o označenom organizme

2.1.15 UC Vlož jedinca

Tento prípad použitia umožňuje používateľovi vložiť do simulácie počas jej behu špecifického jedinca. Vlastnosti tohto jedinca môže používateľ určiť ich zadaním v grafickom rozhraní alebo načítaním daného jedinca zo súboru.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácia je v stave pozastavená

Výstupné podmienky

V simulácií sa nachádza vložený jedinec

Hlavný tok

1. používateľ zvolí možnosť vloženia jedinca do simulácie
2. aplikácia otvorí okno vloženia jedinca do simulácie
3. používateľ v otvorenom okne vloženia jedinca do simulácie vyplní údaje o jedincovi – ide o druhovú príslušnosť jedinca a jeho dna
4. používateľ zvolí možnosť akceptovania nastavení („OK“)
5. aplikácia zatvorí okno vloženia jedinca do simulácie a zmení kurzor myši na kurzor naznačujúci vloženie jedinca; aplikácia zároveň o potrebe umiestnenia jedinca na mapu používateľa dostatočne výrazne upozorní
6. používateľ kliknutím na prázdne políčko mapy vloží na toto políčko vytvoreného jedinca
7. aplikácia vloží a zobrazí nového jedinca na zadanom políčku a zmení kurzor myši do pôvodnej formy

Alternatívny tok 1 – Vloženie jedinca zo súboru

Tento tok umožňuje používateľovi vložiť do simulácie jedinca, ktorý bol v minulosti uložený do súboru. Aktivuje sa namiesto kroku 3 hlavného toku a po jeho vykonaní sa pokračuje krokom 5 hlavného toku.

1. používateľ v okne vloženia jedinca zvolí možnosť načítania zo súboru („Load from file“)
2. aplikácia tvorí okno prehliadača súborov
3. používateľ v otvorenom okne prehliadača súborov nájde súbor s novým jedincom a zvolí ho možnosťou v grafickom rozhraní okna („OK“); používateľ má takisto možnosť okno prehliadača súborov zatvoriť – týmto skončí tento alternatívny tok a pokračuje sa krokom 3 hlavného toku
4. aplikácia skontroluje korektnosť zadaného súboru – ak je validným súborom jedinca pokračuje sa krokom 5 tohto toku, ak nie je validným súborom nastavení druhu upozorní na to používateľa a pokračuje sa krokom 3 tohto toku
5. aplikácia zatvorí okno prehliadača súborov tento alternatívny tok končí

Alternatívny tok 2 – Kontrola správneho vloženia jedinca

Tento tok sa aktivuje po kroku 6 hlavného toku a to v prípade, že používateľ zvolil kliknutím obsadené políčko mapy. Prípad použitia po vykonaní tohto toku pokračuje krokom 6 hlavného toku.

1. aplikácia zobrazí dostatočne výrazné upozornenie informujúce používateľa o tom, že nového jedinca je možné vložiť iba na neobsadené políčko mapy

2.1.16 UC Ulož populáciu

Tento prípad použitia umožňuje používateľovi uložiť do súboru informácie o momentálnej populácii jedného druhu.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácia je v stave pozastavená

Výstupné podmienky

Informácie o populácii zvoleného druhu sú zapísané do zvoleného súboru

Hlavný tok

1. používateľ v grafickom rozhraní zvolí možnosť uloženia populácie
2. aplikácia otvorí okno výberu konkrétneho druhu spomedzi druhov momentálne existujúcich v simulácií
3. používateľ vyberie jeden druh v otvorenom okne výberu druhu a potvrdí svoju voľbu („OK“)
4. aplikácia otvorí okno prehliadača súborov
5. používateľ v okne prehliadača súborov zvolí adresár a meno súboru, v ktorom bude informácia o populácii zvoleného druhu uložená a potvrdí svoju voľbu („OK“)
6. aplikácia vytvorí vo zvolenom adresári súbor so zvoleným menom obsahujúci informácie o momentálnej populácii zvoleného druhu

2.1.17 UC Vlož druh

Tento prípad použitia umožňuje používateľovi vložiť do simulácie počas jej behu špecifický druh organizmov a jeho počiatočnú populáciu. Vlastnosti tohto druhu môže používateľ určiť ich zadaním v grafickom rozhraní alebo načítaním populácie daného druhu zo súboru.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácia je v stave pozastavená

Výstupné podmienky

V simulácií sa nachádza vložený nový druh a jeho populácia

Hlavný tok

1. používateľ zvolí v grafickom rozhraní možnosť vytvorenia nového druhu
2. aplikácia otvorí okno nastavení druhov
3. používateľ zadá ním zvolené parametre druhu do polí na to určených v okne nastavení druhov a zvolí možnosť pridania druhu do zoznamu druhov („Add this species“)
4. aplikácia skontroluje korektnosť zvolených hodnôt parametrov; ak sú korektne zadané, pokračuje sa krokom 5, v opačnom prípade sa pokračuje znovu krokom 3
5. používateľ má možnosť pridať nový druh – pokračuje sa krokom 3, alebo má možnosť pridať doteraz nastavené druhy voľbou v grafickom rozhraní – v tomto prípade sa pokračuje krokom 6
6. aplikácia zatvorí okno nastavení druhov

Alternatívny tok 1 – Načítanie populácie druhu zo súboru

Tento tok umožňuje používateľovi načítať populáciu druhu zo súboru. Aktivuje sa namiesto kroku 3 hlavného toku a po jeho vykonaní sa pokračuje krokom 5 hlavného toku.

1. používateľ zvolí možnosť načítania populácie druhu zo súboru („Load from file“)
2. aplikácia otvorí okno prehliadača súborov
3. používateľ v otvorenom okne prehliadača súborov nájde súbor s populáciou druhu a zvolí ho možnosťou v grafickom rozhraní okna („OK“); používateľ má takisto možnosť okno prehliadača súborov zatvoriť – týmto skončí tento alternatívny tok a pokračuje sa krokom 3 hlavného toku
4. aplikácia skontroluje korektnosť zadaného súboru – ak je validným súborom populácie druhu pokračuje sa krokom 5 tohto toku, ak nie je validným súborom nastavení druhu upozorní na to používateľa a pokračuje sa krokom 3 tohto toku
5. aplikácia zatvorí okno prehliadača súborov a do okna nastavení druhu pridá druh načítaný zo zadaného súboru

2.1.18 UC Ulož aktuálny stav simulácie

Tento prípad použitia umožňuje používateľovi uložiť do súboru momentálny stav simulácie. Tento súbor potom môže poslúžiť na pokračovanie tejto simulácie v neskoršom čase alebo na inom stroji.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácia je v stave pozastavená

Výstupné podmienky

Momentálny stav simulácie je uložený v súbore

Hlavný tok

1. používateľ volí v grafickom rozhraní možnosť uloženia stavu simulácie
2. aplikácia otvorí okno prehliadača súborov
3. používateľ v otvorenom okne prehliadača súborov vyberie adresár a meno súboru, v ktorom bude informácia o momentálnom stave simulácie uložená
4. aplikácia do zvoleného adresára a súboru zapíše momentálny stav simulácie

2.1.19 UC Vlož objekt

Tento prípad použitia umožňuje používateľovi vložiť do sveta simulácie isté objekty mapy. Ide o vloženie prekážok a základnej potravy.

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácia je v stave pozastavená

Výstupné podmienky

V simulácií sa nachádzajú vložené nové objekty

Hlavný tok

1. používateľ v grafickom rozhraní zvolí jednu z možností vloženia objektu (vloženie potravy/prekážky)
2. aplikácia zmení kurzor myši na kurzor symbolizujúci vkladanie objektov na mapu a dostatočne výrazne upozorní používateľa na potrebu samotného vloženia objektu na mapu
3. používateľ kliknutím ľavého tlačidla myši na prázdne políčko mapy vloží zvolený typ objektu na mapu

4. ak chce používateľ pokračovať vo vkladaní ďalších objektov, pokračuje sa krokom 3; ak nechce pokračovať klikne na mapu pravým tlačidlom myši a prípad použitia končí

Alternatívny tok 1 – Kontrola správneho vloženia objektu

Tento tok sa aktivuje po kroku 3 hlavného toku a to v prípade, že používateľ zvolil kliknutím obsadené políčko mapy. Prípad použitia po vykonaní tohto toku pokračuje krokom 4 hlavného toku.

1. aplikácia zobrazí dostatočne výrazné upozornenie informujúce používateľa o tom, že nový objekt je možné vložiť iba na neobsadené políčko mapy

2.1.20 UC Odstráň objekt

Tento prípad použitia umožňuje používateľovi odstrániť zo sveta simulácie isté objekty. Ide o možnosť odstránenia prekážok a potravy

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, simulácia je v stave pozastavená

Výstupné podmienky

Zo simulácie sú odstránené zvolené objekty

Hlavný tok

1. používateľ v grafickom rozhraní zvolí možnosť odstránenia objektu (zahŕňa aj odstránenie potravy aj prekážky)
2. aplikácia zmení kurzor myši na kurzor symbolizujúci odstraňovanie objektov z mapy a dostatočne výrazne upozorní používateľa na potrebu samotného odstránenia objektov z mapy
3. používateľ kliknutím ľavého tlačidla myši na políčko mapy obsahujúce základnú potravu alebo prekážku tento objekt odstráni z mapy; ak používateľ klikne na políčko mapy obsahujúce iný objekt (alebo prázdne políčko) aplikácia dostatočne výrazne zobrazí upozornenie, že odstrániť je možné len základnú potravu a prekážky
4. ak chce používateľ pokračovať v odstraňovaní ďalších objektov, pokračuje sa krokom 3; ak nechce pokračovať klikne na mapu pravým tlačidlom myši a prípad použitia končí

2.1.21 UC Vytvor mapu

Tento prípad použitia umožňuje používateľovi vytvoriť pre simuláciu mapu pomocou editora máp. Tento prípad použitia je zahrnutý v prípade použitia „Nastav parametre sveta“ (bližšie opísaný v kapitole 2.1.2 „UC Nastav parametre sveta“).

Aktéri

Používateľ

Vstupné podmienky

Používateľ spustil aplikáciu, v aplikácií je otvorené okno nastavení sveta

Výstupné podmienky

Je vytvorený súbor obsahujúci novovytvorenú mapu a táto mapa je nastavená ako momentálna mapa sveta simulácie

Hlavný tok

1. používateľ v okne nastavení sveta zvolí možnosť vytvorenia mapy pomocou editora
2. aplikácia otvorí okno editora máp
3. používateľ klikaním a ťahaním myši v mriežke zobrazenej v okne editora máp vytvorí mapu s prekážkami podľa svojej voľby
4. používateľ zvolí v editore máp možnosť uloženia mapy do súboru
5. aplikácia otvorí okno prehliadača súborov
6. používateľ v otvorenom okne prehliadača súborov vyberie adresár a meno súboru, v ktorom bude mapa uložená
7. aplikácia do zvoleného adresára a súboru zapíše mapu
8. aplikácia v nastaveniach sveta nastaví novovytvorenú mapu ako aktívne nastavenie a toto nastavenia zobrazí v okne nastavení sveta

2.2 Nefunkcionálne požiadavky

Základnou požiadavkou na vytváraný systém je, aby bol vytvorený ako desktopová aplikácia. Nie je totiž potrebné, aby sme použili architektúru klient-server, keďže moderné počítače umožňujú zvládať beh simulácie aj vizualizácie vo viacerých vláknach v pre nás prijateľnom čase.

Dôležitou požiadavkou je výkon systému. Je potrebné zabezpečiť aby aplikácia vedela dostatočne rýchlo vykonať simuláciu a vizualizáciu sveta HERBAL. Dostatočná rýchlosť v tomto prípade znamená schopnosť systému plynulo (cca. 25 tikov za sekundu) simulovať a zobrazovať momentálny stav sveta minimálne o veľkosti 1000x1000 políčok, pri neobmedzenom počte organizmov. Pre naplnenie tejto požiadavky je dôležité zvoliť vhodný programovací jazyk.

Používateľské rozhranie musí byť vysoko intuitívne, aj napriek tomu, že by malo umožňovať nastavovať veľké množstvo parametrov.

Ďalšou požiadavkou na náš produkt je jeho stabilita. Simulácie sa budú vykonávať aj niekoľko hodín, je preto kritické aby program počas tejto doby spoľahlivo fungoval a v žiadnom prípade neukončil svoj beh.

Nami vytváraná aplikácia by mala byť jednoducho rozšíriteľná. Ide najmä o rozšíriteľnosť na úrovni kódu v podobe pridávania nových typov akcií a podmienok, vlastností organizmov a sveta a podobne. V zadaní sa síce požaduje DNA organizmu zložená z pravidiel, ale aplikácia by mala byť vytvorená tak, aby sa dala pravidlová DNA vymeniť za inú bez väčších ťažkostí.

3 Návrh

Po analýze problémovej oblasti sme sa rozhodli pre implementovanie aplikácie objektovo-orientovaným prístupom. Zvolili sme si jazyk C#, ktorý nám bol odporučený počas konzultácií s vedúcim projektu a s externým expertom (Ing. Michal Barla). Dôvodom tejto voľby (a dôvodom na uprednostnenie C# pred jazykom Java) bola možnosť poskytnutia väčšej výpočtovej rýchlosti vykonávania programu implementovaného v jazyku C#.

Po zvolení implementačného jazyka C# sme sa rozhodli používať na implementovanie vizualizácie simulácií framework Microsoft XNA pre C#. Tento framework sme zvolili z dôvodu veľkej dostupnosti kvalitnej dokumentácie a návodov na internete, ktoré nám umožnia vytvoriť v tomto frameworku kvalitnú vizualizáciu. Framework XNA však zároveň prináša do aplikácie jedno obmedzenie – obsahuje len knižnice pre 32-bitové systémy a tým pádom aplikácia nedokáže využiť potenciál 64-bitových systémov.

Vývojové prostredie, ktoré budeme používať bude Microsoft Visual Studio 2008.

3.1 Návrh riadenia a modelu systému

V tejto časti návrhu opisujeme časť systému, ktorá je zodpovedná za riadenie a aplikačnú logiku programu. Dokumentáciu tejto časti návrhu vypracoval Ján Romaňák.

3.1.1 Architektúra systému

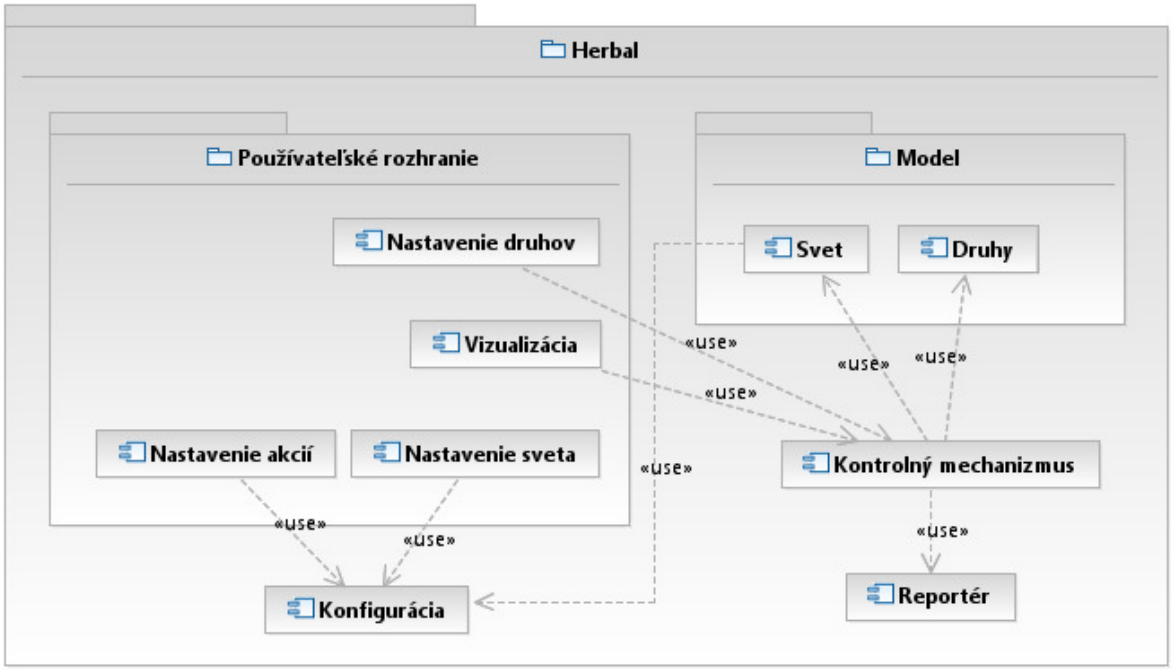
Na základe analýzy problému sme identifikovali štyri hlavné moduly, na ktoré aplikáciu rozdelíme. Tieto moduly sú:

- model sveta – tvorí samotný simulačný mechanizmus, pozostáva z týchto hlavných entít:
    - druhy – skupina/hierarchia tried, ktoré budú reprezentovať jednotlivé organizmy a druhy organizmov; tieto triedy budú charakterizovať ich správanie (gény)
    - svet – tieto triedy budú reprezentovať svet v podobe mriežky, budú umožňovať existenciu a pohyb organizmov v priestore
- kontrolný mechanizmus – tieto triedy budú zodpovedné za riadenie simulácie – t.j.:
    - inicializáciu a spravovanie druhov a sveta
    - spúšťanie a zastavovanie simulácie
    - aplikovanie mechanizmu času na svet
- používateľské rozhranie – triedy umožňujúce používateľovi monitorovať priebeh simulácií umelého života a prípadne do nich robiť zásahy, zmeny parametrov

- vizualizácia – dôležitá súčasť rozhrania, ktorá umožní prehľadné sledovanie simulácií a prípadnú interakciu používateľa so svetom

- reportovanie – triedy zabezpečujúce zaznamenávanie priebehu simulácií; tieto informácie umožnia vykonávanie analýzy dát na základe zvolených zachytených štatistických údajov

Na obrázku (Obr. 4) je znázornený diagram komponentov, ktorý vyjadruje túto architektúru.

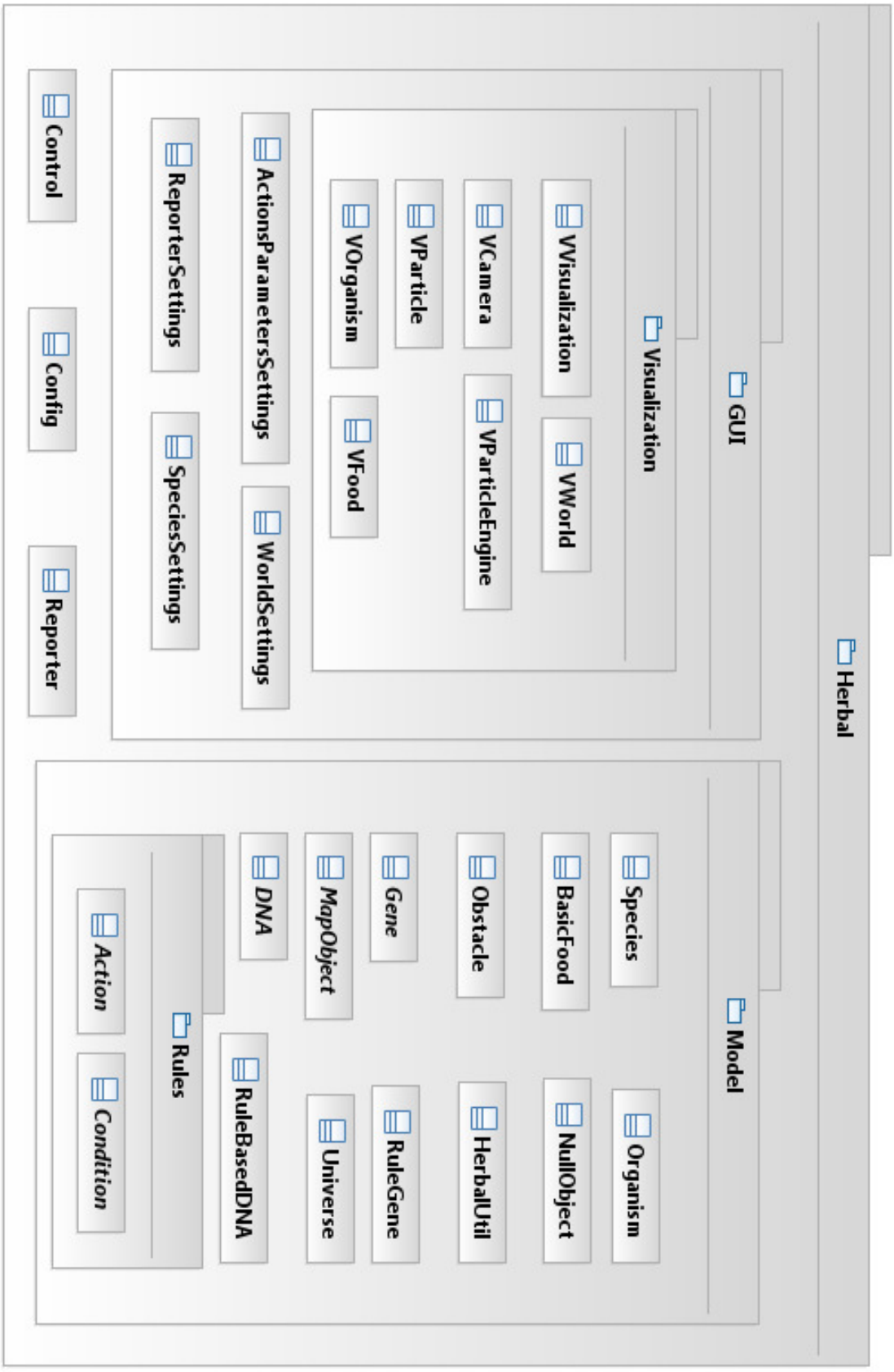


Obr. 4 - Architektúra systému

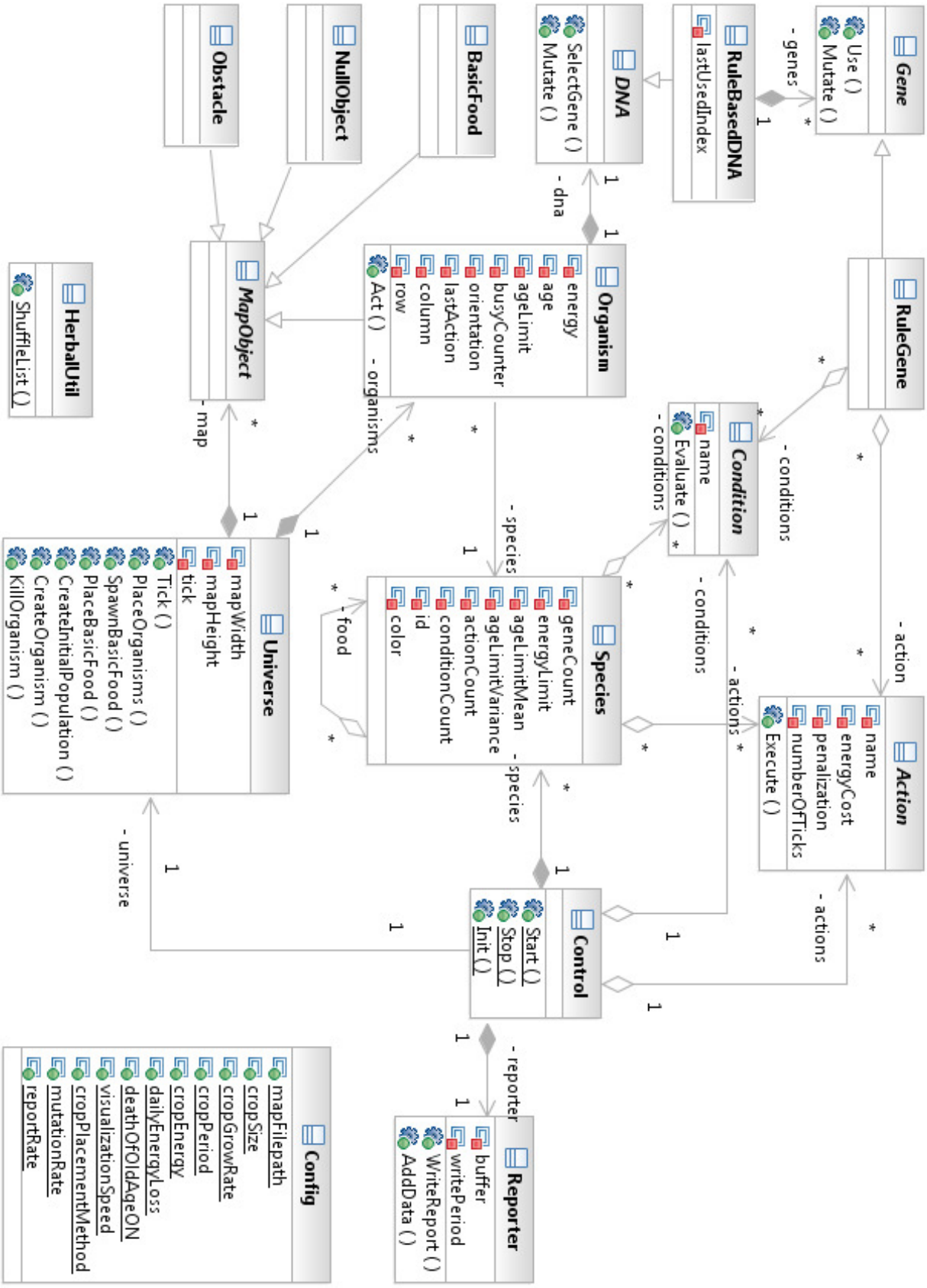
3.1.2 Model tried

V tejto podkapitole sa nachádza nami navrhovaná dekompozícia systému na triedy. Pri návrhu tejto dekompozície sme vychádzali z kapitoly 3.1.1 „Architektúra systému“ a sústredili na to, aby bol systém ľahko rozšíriteľný. Na obrázku Obr. 5 je znázornený diagram balíkov, ktorý znázorňuje rozdelenie tried v súlade s architektúrou systému. Atribúty a metódy tried spolu s vzťahmi medzi triedami sú znázornené na diagramoch tried na obrázkoch Obr. 6 a Obr. 7. Rozšíriteľnosť sme dosiahli zakomponovaním abstraktných tried *DNA*, *Gene*, *Condition*, *Action* a *MapObject*. V nasledujúcej časti kapitoly sa nachádza popis jednotlivých navrhnutých tried.

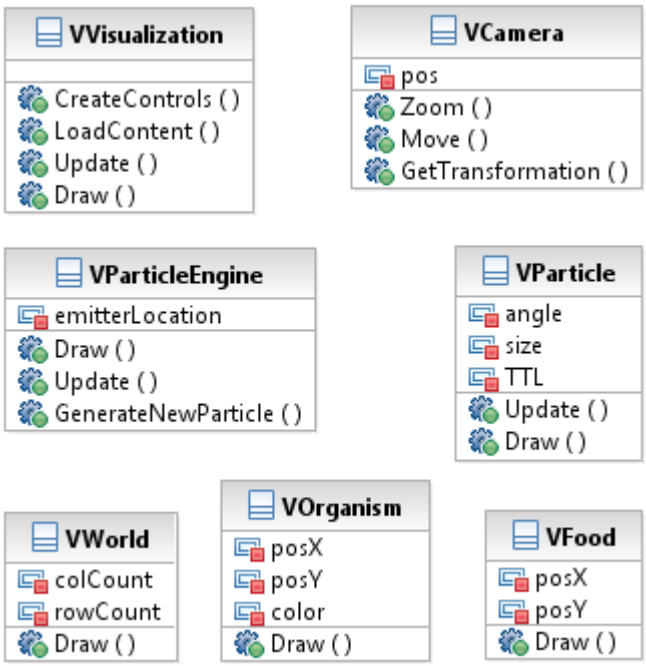
Triedy (prípadne ich odvodené podtriedy v prípade abstraktných tried) *NullObject*, *Obstacle*, *BasicFood*, *Condition* a *Action* sú navrhnuté ako Singleton-y pre pamäťové zefektívnenie programu – aby nebolo potrebné ukladať veľké množstvo vytvorených objektov týchto typov.



Obr. 5 - Diagram balíkov



Obr. 6 – Diagram tried jadra systému



Obr. 7 - Diagram tried vizualizácie

Control

Táto trieda reprezentuje základný vstupný bod vytvorenia modelu. Obsahuje statické metódy *Init()*, *Start()* a *Stop()*. Metóda *Init()* slúži na vytvorenie sveta (*Universe*), vytvorenie mechanizmu logovania dát (*Reporter)* a pridanie druhov a počiatočných populácií týchto druhov organizmov. Metódy *Start()* a *Stop()* slúžia na spúšťanie a zastavovanie simulácie v modeli.

Reporter

Slúži na zapisovanie štatistických informácií o simulácií. Metódou *AddData()* sa mu posielajú informácie z *Universe*, pričom tieto sú spracované pomocou vyrovnávacej pamäte a zapísané do súboru metódou Write*Report().*

Species

Reprezentuje konkrétny druh organizmov. Obsahuje údaje o konkrétnom druhu: počet génov v DNA, počet podmienok a akcií v génoch, maximálny limit energie jedincov daného druhu a parametre ich možného maximálneho veku. Každý druh takisto obsahuje zoznam tých druhov, ktoré sú jeho potravou – takto je možné jednoducho definovať vzťah korisť-predátor medzi druhmi.

Universe

Predstavuje mapu sveta. To znamená, že obsahuje dvojrozmerné pole reprezentujúce mapu, v ktorom sa nachádzajú objekty typu *MapObject*. *Universe* zároveň vykonáva kontrolu nad týmito objektmi – vytvára napríklad počiatočné populácie (*CreateInitialPopulation()*) a je zodpovedný aj za odstraňovanie organizmov z mapy (*KillOrganism()*). Takisto obsahuje zoznam momentálne žijúcich organizmov *organisms*. Jednou z úloh *Universe* je aj vytváranie novej potravy *BasicFood* v stanovených okamihoch (každých niekoľko kôl – parameter v triede *Config*) simulácie. Keďže

Universe obsahuje všetky tieto informácie, je aj počiatočným centrom vytvárania záznamov štatistických dát – obsahuje štruktúru *ReportData*, ktorá obsahuje informácie o momentálnom stave sveta.

Najdôležitejšou funkciou *Universe* je však vykonávanie jeho metódy *Tick()*, ktorá v náhodnom poradí dáva príležitosť organizmom vykonať svoje akcie v každom kole.

MapObject

Táto abstraktná trieda reprezentuje objekty umiestnené na mape. Slúži iba na to, aby objekty na mape (organizmy, prekážky, potrava) mali možnosť dediť od jedného typu a mohli byť tak jednoducho reprezentované v mape v *Universe*.

NullObject

Konkrétna trieda odvodená od *MapObject*. Reprezentuje voľné políčko na mape.

Obstacle

Konkrétna trieda odvodená od *MapObject*. Reprezentuje políčko na mape, na ktorom sa nachádza prekážka.

BasicFood

Konkrétna trieda odvodená od *MapObject*. Reprezentuje základný druh potravy pre všetky druhy organizmov. Energia poskytnutá organizmu po skonzumovaní tejto potravy je nastavovaná cez *Config.cropEnergy*.

Organism

Konkrétna trieda odvodená od *MapObject*. Reprezentuje organizmus. Obsahuje rôzne informácie o konkrétnom organizme: momentálna energia (*energy*), orientácia vo svete (*orientation*), súradnice na mape (*row*, *column*), informáciu o tom, či práve organizmus vykonáva nejakú dlhodobú akciu (*busyCounter*) a maximálny vek, ktorý môže daný organizmus/jedinec dosiahnuť (*ageLimit*).

Organism takisto obsahuje odkazy na objekt svojho druhu *species* a na svoju DNA *dna*. Tieto charakterizujú jeho správanie, ktoré vykonáva pomocou funkcie *Act()*.

Organism pomocou troch rôznych konštruktorov umožňuje vytvoriť náhodný organizmus, organizmus naklonovaný a zmutovaný z jedného konkrétneho organizmu a organizmus vytvorený skrížením a zmutovaním dvoch konkrétnych organizmov.

DNA

Abstraktná trieda predstavujúca dna/súbor génov organizmu. Deklaruje metódu *SelectGene()* pre svoje podtriedy. Táto metóda vyjadruje výber jedného z mnohých génov v dna na to, aby bol aplikovaný pri vykonaní akcie.

RuleBasedDNA

Konkrétna trieda odvodená od *DNA*. Reprezentuje dna, ktorá je založená na pravidlových génoch – t.j. génoch obsahujúcich pravidlá v tvare *„Podmienka → Akcia“.* Obsahuje zoznam génov *genes* a implementáciu metódy *SelectGene()*, ktorá vyberá jeden z génov na vykonanie podľa nastavenej metódy (*Species.geneSelectionMethod*).

RuleBasedDNA obsahuje tri rôzne konštruktory, ktoré organizmu umožňujú vytvoriť dna náhodnú v rámci možností jeho druhu, dna zmutovanú z dna jeho jedného rodiča (klonovanie) a dna skríženú a zmutovanú z dna jeho dvoch rodičov (kríženie).

Gene

Abstraktná trieda predstavujúca jeden gén ako časť dna. Deklaruje metódu *Use()* pre svoje podtriedy. Táto metóda vyjadruje aplikovanie/vykonanie daného génu.

RuleGene

Konkrétna trieda odvodená od *Gene*. Reprezentuje gén vytvorený z podmienok a akcií v tvare *„Podmienka → Akcia“. RuleGene* obsahuje zoznam podmienok *conditions* a zoznam akcií *actions*. Implementuje metódu *Use()*, ktorá najskôr vyhodnotí všetky podmienky a ak všetky platia vykoná postupne všetky akcie (pokiaľ niektorá z nich neskončí neúspechom neumožňujúcim vykonať po nej nasledujúce akcie – napríklad smrťou organizmu).

RuleGene obsahuje dva rôzne konštruktory – pre vytvorenie pravidlového génu využívajúceho podmienky a akcie dostupné danému druhu a pre vytvorenie génu z génov dvoch rodičov pri krížení.

Condition

Abstraktná trieda reprezentujúca podmienku v pravidlovom géne. Deklaruje pre svoje odvodené triedy premennú *name* obsahujúcu meno podmienky (tak ako bude zobrazené v používateľskom rozhraní) a metódu *Evaluate()*, ktorá vyhodnotí platnosť danej podmienky.

Základná skupina konkrétnych navrhnutých podmienok odvodených od tejto triedy je uvedená v kapitole 3.1.3 „Návrh konkrétnych podmienok a akcií“.

Action

Abstraktná trieda reprezentujúca akciu v pravidlovom géne. Deklaruje pre svoje odvodené triedy premennú *name* obsahujúcu meno akcie (tak ako bude zobrazené v používateľskom rozhraní) a metódy *Execute()* a *ApplyEffect()*. *Execute()* vykonáva samotnú akciu – t.j. napríklad robí zmeny v *Universe* a *ApplyEffect()* aplikuje efekty akcie ako sú odčítanie energie organizmu a nastavenie *Organism.busyCounter* pri dlhodobých akciách.

Action takisto obsahuje parametre *energyCost*, *penalization*, *numberOfTicks*, ktoré predstavujú cenu akcie, penalizáciu pri jej neúspešnom vykonaní (takisto v jednotkách energie) a dobu trvania akcie.

Základná skupina konkrétnych navrhnutých akcií odvodených od tejto triedy je uvedená v kapitole 3.1.3 „Návrh konkrétnych podmienok a akcií“.

VVisualization

Konkrétna trieda reprezentujúca jadro vizualizačnej časti, ktorá sa stará o réžiu samotnej vizualizácie. Je odvodená od triedy Game, ktorá je súčasťou frameworku XNA. Táto trieda obsahuje metódu Draw(), v ktorej renderuje všetko, čo sa má zobraziť na obrazovke (t.j. svet, organizmy, potrava, ovládací panel s ovládacími prvkami). Táto trieda takisto obsahuje metódu Update(), v ktorej kontroluje zmeny, ktoré sa udiali vo vizualizácií v čase. Vďaka nej je tak možné implementovať rôzne animácie, alebo vstupy od používateľa.

V tejto triede sa takisto tvorí ovládací panel vizualizačnej časti. Všetky ovládacie prvky tohto panela sú vytvorené v metóde CreateControls().

VCamera

Konkrétna trieda reprezentujúca kameru v dvojrozmernom svete vizualizácie. Vďaka nej je možné sa po scéne pohybovať (metóda Move()), či scénu približovať a odďaľovať (metóda Zoom()).

Pri renderovaní všetkých objektov je potrebné vypočítať momentálnu pozícu kamery. O toto sa stará metóda GetTransformation(), ktorá maticovými výpočtami zaistí vždy správne nastavenie pozície kamery.

VParticleEngine

Konkrétna trieda, ktorá ma na starosti réžiu časticových systémov. Obsahuje zoznam častíc, z ktorých sa celý časticový systém skladá a nad týmito časticami vykonáva rôzne operácie, pričom sa stará o generovanie nových častíc, o likvidáciu častíc, ktorým vypršal čas žitia, ako aj o smer animácie týchto častíc.

VParticle

Konkrétna trieda, ktorá reprezentuje jednu konkrétnu časticu, ktorá sa môže použiť ako súčasť veľkého časticového systému. Ku každej častici je potrebné nastaviť dobu žitia, veľkosť, farbu, či textúru.

Častice sa používajú v časticových systémoch na zobrazovanie rôznych efektov a slúžia na spestrenie vizualizačnej časti.

VWorld

Táto trieda reprezentuje svet Herbalu. Vďaka metóde Draw() vykresľuje na scénu celý svet, pričom prechádza políčko po políčku a zisťuje, či sa na danom mieste nachádza, alebo nenachádza prekážka. Prekážka, ako aj prázdne políčko majú definovanú jednoznačnú textúru.

VOrganism

Trieda reprezentujúca vizualizáciu herba. Obsahuje metódu Draw(), vďaka ktorej vykreslí na danú pozíciu herba so zadanou textúrou. Orientáciu herba je možné zmeniť zmenou textúry. Je možné v systéme rozlišovať viac druhov herbov, preto táto trieda obsahuje atribút color, ktorý reprezentuje farbu, ktorá je jedinečná pre každý druh herbov.

VFood

Konkrétna trieda reprezentujúca vizualizáciu potravy. Vďaka metóde Draw() vykreslí na danú pozíciu objekt so zadanou textúrou.

3.1.3 Návrh konkrétnych podmienok a akcií

V tejto kapitole uvádzame základnú množinu podmienok a akcií, ktoré budú súčasťou modelu sveta. Tieto podmienky a akcie predstavujú špecifikácie pre konkrétne triedy, ktoré sú odvodené od tried „Condition“ a „Action“, ktoré sú uvedené v kapitole 3.1.2 „Model tried“.

Vytvorili sme návrh týchto základných akcií:

- *AlwaysTrue* – Predstavuje prázdnu podmienku – žiaden senzor. Vždy sa automaticky vyhodnocuje ako pravdivá. Ak sa nachádza v jednom géne organizmu spolu s inými podmienkami efektívne skracuje podmienkovú časť génu o jednu podmienku.
- *AlwaysFalse* – Vždy sa vyhodnocuje ako nesplnená, teda natrvalo blokuje gén, v ktorom sa nachádza.
- *SeeFood* – Táto podmienka je splnená, ak sa priamo pred daným organizmom na mape sveta nachádza políčko na ktorom je potrava (objekt triedy „BasicFood“ –trieda bližšie opísaná v kapitole 3.1.2 „Model tried“).
- *SeeFreeTile* – Táto podmienka je splnená, ak sa na mape sveta priamo pred daným organizmom nachádza voľné políčko potrava (objekt triedy „NullObject“ –trieda bližšie opísaná v kapitole 3.1.2 „Model tried“).
- *SeeObstacle* – Táto podmienka sa splní ak sa priamo pred daným organizmom nachádza na mape sveta prekážka/okraj sveta potrava (objekt triedy „Obstacle“ –trieda bližšie opísaná v kapitole 3.1.2 „Model tried“).
- *SeeFriend* – Táto podmienka sa vyhodnotí ako pravdivá, ak sa priamo pred daným organizmom na mape sveta nachádza iný organizmus rovnakého druhu.
- *SeeAlien* – Táto podmienka sa vyhodnotí ako pravdivá, ak sa priamo pred daným organizmom na mape sveta nachádza iný organizmus odlišného druhu než tento organizmus.
- *SeeOrganism* – Táto podmienka sa vyhodnotí ako pravdivá, ak sa priamo pred daným organizmom nachádza na mape sveta organizmus ľubovoľného druhu.

Vykonali sme návrh týchto základných akcií:

- *Null* – Ide o prázdnu akciu, teda akciu, ktorá nič nevykoná. Nespotrebuje sa pri nej žiadna energia a neminie sa žiadny čas, pretože táto akcia je preskočená. Ak sa táto akcia nachádza v géne organizmu a vyberie sa na aplikovanie, organizmus už v danom kole nevykonáva ďalší ťah. Ak sa nachádza v jednom géne spolu s inými akciami efektívne skracuje akciovú časť génu o jednu akciu.

- *Wait* – Reprezentuje akciu čakania, táto akcia sa vykoná, ale organizmus nerobí nič, len čaká. Na túto akciu organizmus spotrebuje relatívne menej energie než na iné komplikovanejšie akcie.

- *TurnLeft* – Pri vykonaní tejto akcie sa organizmus na mieste otočí o 90° vľavo, teda proti smeru hodinových ručičiek. Po vykonaní tejto akcie organimzus ostáva na rovnakom políčku mapy sveta ako pred jej vykonaním. Akcia by mala byť vykonateľná kedykoľvek a spotrebuje sa na ňu relatívne malé množstvo energie a trvá spravidla len jeden tik.

- *TurnRight* – Pri vykonaní tejto akcie sa organizmus na mieste otočí o 90° vpravo, teda v smere hodinových ručičiek. Po vykonaní tejto akcie organizmus ostáva na rovnakom políčku ako pred jej vykonaním. Akcia by mala byť vykonateľná kedykoľvek a spotrebuje sa na ňu malé množstvo energie a trvá spravidla len jeden tik.

- *Move* – Pri vykonaní tejto akcie sa organizmus pohne na mape sveta vpred o jedno políčko v tom smere, v ktorom je práve otočený. Na túto akciu herb spotrebuje časť energie a minie časť času bez ohľadu na to, či sa skončila úspechom. Túto akciu je možné vykonať len v prípade, keď sa priamo pred organizmom nachádza prázdne políčko. Ak sa pred organizmom nachádza iný typ políčka, akcia bude vykonaná neúspešne – organizmus teda zostáva na svojom pôvodnom mieste na mape sveta a bude mu odpočítaná energia rovná hodnote parametra penalizácie pre túto akciu.

- *Clone* – Pri vykonaní tejto akcie sa organizmus rozdelí na dvoch jedincov, pričom sa spotrebuje v porovnaní s ostatnými akciami väčšie množstvo času a energie. Nový jedinec sa vytvorí s DNA totožnou s DNA pôvodného jedinca, jeho DNA sa ale následne zmutuje podľa parametrov nastavených pre mutáciu. Ak chce organizmus vykonať túto akciu, musí mať aspoň isté stanovené množstvo energie a musí mať vedľa seba aspoň jedno políčko mapy voľné. Novovzniknutý organizmus „sa narodí“ na náhodnom voľnom políčku pri pôvodnom organizme. Postup samotnej mutácie je bližšie opísaný v kapitole 3.1.4 „Návrh ďalších aspektov simulácie“.

- *Cross* – Predstavuje kríženie dvoch organizmov, ktoré ak je úspešné, tak vznikne nový organizmus. Pri tejto akcií sa spotrebuje v porovnaní s ostatnými akciami väčšie množstvo času a energie. Aby bolo možné vykonať kríženie, musí sa vedľa organizmu na mape sveta nachádzať iný organizmus rovnakého druhu, musí byť vedľa organizmu na mape voľné aspoň jedno políčko a obidva organizmy musia mať istú minimálnu energiu a vek. Po úspešne vykonanom krížení sa na náhodnom voľnom políčku mapy vedľa organizmu „narodí“ nový jedinec, ktorý získa od rodičov jeho počiatočnú energiu a skrížený chromozóm, ktorý bude ešte aj zmutovaný podľa parametrov mutácie. Postup samotného kríženia je bližšie opísaný v kapitole 3.1.4 „Návrh ďalších aspektov simulácie“.

3.1.4 Návrh ďalších aspektov simulácie

Časový mechanizmus

Priebeh simulácie je delený na tzv. tiky. V každom tiku vykonávajú všetky organizmy svoje akcie. Akcie môžu trvať jeden alebo viac tikov, pričom každý organizmus si najprv vyberie gén, ktorého

podmienky sú splnené a potom začne vykonávať akcie, ktoré vybraný gén obsahuje. Následok všetkých akcií sa prejaví okamžite, pričom organizmus v nasledujúcich tikoch musí čakať. Počet týchto tikov je určený sumou trvania vykonaných akcií. Keď organizmu vyprší doba, ktorú musí čakať, vyberá si nový gén, ktorý použije.

Výber Organizmov pri vykonávaní ich akcií

Organizmy sa vyberajú v náhodnom poradí nezávisle od polohy a druhu.

Výber génu

Výber génu sa vykonáva na základe režimu, ktorý bol používateľom vybratý pre druh. Gén sa vyberá až dovtedy, kým sa nevyberie gén ktorý má splnené všetky podmienky alebo kým počet pokusov nedosiahne maximálny počet pokusov.

Možné režimy výberu sú:

- Náhodný výber – vyberie sa náhodný gén z DNA
- Sekvenčný výber – je uložené poradové číslo génu, ktorý bol vybraný ako posledný a vždy sa vyberá nasledujúci po ňom. Keď bol naposledy vybraný posledný gén DNA, začne sa zase od prvého génu.

Mutácia DNA

Pri mutácii DNA sa určí náhodný počet génov, ktoré sa majú mutovať. Tento počet nepresahuje stupeň mutácie (nastavuje sa pri konfigurácii sveta). Potom sa mutujú náhodne vybrané gény z DNA v určenom počte. Ďalšie aspekty mutácie sú opísané pri popise akcií *Clone* a *Cross* v kapitole 3.1.3 „Návrh konkrétnych podmienok a akcií“.

Mutácia génu

Pri mutácii génu sa určí náhodný počet častí génu (podmienok a akcií), ktoré sa majú mutovať. Tento počet nepresahuje stupeň mutácie (nastavuje sa pri konfigurácii sveta). Potom sa náhodne vybrané časti génu v určenom počte vymenia za nové náhodne vybrané časti tak, aby sa zhodoval ich typ (podmienky sa menia za podmienky a akcie za akcie). Podmienky aj akcie sa vyberajú z množiny podmienok a akcií pre daný druh organizmu.

Kríženie DNA

Pri krížení DNA sa množiny génov spoja do jednej množiny. Potom pri vytváraní novej DNA pre potomka sa vyberajú náhodne gény z tejto množiny tak, že každý gén môže byť vybraný len raz. Ďalšie aspekty kríženia sú opísané pri popise akcie *Cross* v kapitole 3.1.3 „Návrh konkrétnych podmienok a akcií“.

Rast potravy

Potrava sa objavuje na mape vždy na začiatku sezóny, ktorej dĺžka je určená používateľom pri konfigurácii sveta. Potrava sa objavuje vždy na náhodných voľných políčkach mapy. Množstvo

potravy, ktoré sa na mape objaví je závislé od zvoleného režimu pri konfigurácii sveta. Možné režimy sú:

- Rovnaké množstvo v každej sezóne – na začiatku každej sezóny sa objaví rovnaké množstvo potravy. Toto množstvo určuje používateľ pri konfigurácii sveta.
- Doplnok do konštantného množstva – Na začiatku každej sezóny sa objaví také množstvo potravy, aby sa celkový počet potravy vo svete rovnal počtu nastavenému pri konfigurácii sveta.
- Odvodené od aktuálneho množstva potravy na svete – pri konfigurácii sveta používateľ určí aký násobok aktuálneho množstva potravy sa má objaviť.

3.1.5 Návrh reportéra

Reportér slúži na ukladanie štatistík simulácie(reportov) do súboru. Reporty sa budú tvoriť po každých n tikoch v závislosti od nastavenia používateľom. Report bude vždy odrážať iba obdobie od vytvorenia posledného reportu.

Samotný report sa bude tvoriť nasledovným spôsobom. Control za n tikov zavolá funkciu GetReportData() z triedy Universe ktorá vracia hashovaciu tabuľku s vytvorenými reportami. Následne prijaté dáta pošle triede Reporter ako parameter funkcie AddData() a tá s nimi ďalej pracuje. Trieda Reporter zabezpečuje ukladanie reportérov do pomocnej pamäte (buffer-u) a realizuje ich zápis do súboru pomocou funkcie WriteData(). Zápis sa realizuje až po prijatí 100 sád reportov, alebo po zastavení simulácie. V prototype sa počíta so štandardným zápisom do súboru output.txt, ktorý sa po spustení programu vytvorí, ak neexistuje, ak existuje tak sa vymaže jeho predchádzajúci obsah. V plnej verzii programu sa počíta s nastavením názvu súboru a výberom umiestnenia súboru používateľom.

V prototype sa počíta iba so zápisom základných štatistík, ktoré sú nasledovné:

- číslo tiku v ktorom sa daný report vytvoril
- veľkosť populácie
- počet potravy vo svete
- priemerná energia populácie
- priemerný vek populácie
- priemerný vek dožitia populácie

Pre každý report bude vytvorená samostatná funkcia, ktorá bude vracať výsledok daného reportu. Vo finálnej verzii programu si používateľ bude môcť sám vybrať, ktoré reporty chce ukladať do súboru, počíta sa tiež s rozšírením sady reportov o nasledovné:

- sada reportov zahŕňajúca ďalšie populačné štatistiky (priemerný prírastok populácie, počet nový jedincov, počet umretých jedincov..)

- sada reportov vyhodnocujúca akcie (počet vykonaní jednotlivých akcií, úspešné, neúspešné akcie, pomer jednotlivých akcií..)
- sada reportov vyhodnocujúca genóm jedincov (početnosť jednotlivých akcií, podmienok, funkčných génov..)
- reporty zamerané na jednotlivé druhy organizmov (počet jedincov druhu, priemerná energia druhu..)

3.2 Návrh vizualizácie

V tejto časti návrhu opisujeme vizualizáciu systému a grafické používateľské rozhranie. Túto časť vypracovali Miroslav Mikuláš, Miloš Auder a Lukáš Ďurčák.

3.2.1 Model tried

Vizualizačná časť má na starosti zobrazovanie sveta v reálnom čase. Samotná vizualizácia je volaná z hlavnej časti programu, pričom prekresľovanie scény prebieha v rovnakom časovom intervale ako simulačný tik..

Vizualizácia je samostatná časť programu, pričom informácie, ktoré potrebuje, čerpá z hlavnej časti programu. Pre bezproblémový chod vizualizácie sú potrebné aspoň nasledujúce údaje:

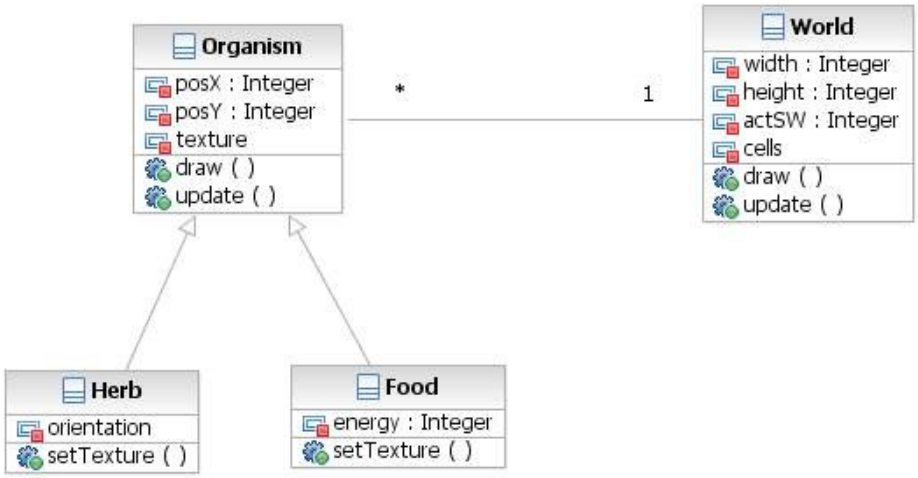
- rozmery sveta
- pozície stien
- pozície všetkých herbov
- pozície všetkých potráv
- orientácia všetkých herbov

Vďaka týmto údajom je možné vizualizovať stav sveta v každom jednom časovom okamihu. Samotný organizmus je reprezentovaný triedou Organism. Z tejto triedy sú odvodené triedy Herb a Food, pričom trieda Herb reprezentuje herba a trieda Food reprezentuje potravu. Každý organizmus má jedinečnú pozíciu vo svete, ktorá je vyjadrená atribútmi posX a posY. Aby boli organizmy na pohľad rozoznateľné, to znamená aby používateľ odlíšil herba od potravy, je potrebné aby mal organizmus svoju textúru. Organizmus sa vykreslí na obrazovku procedúrou draw(). Procedúra update () sa stará o zmenu polohy organizmu, zmeny textúry, prípadne aj zničenie organizmu. Trieda, ktorá reprezentuje herba má navyše pridanú orientáciu, do ktorého smeru je herb otočený. Na základe tejto vlastnosti sa vyberie tá správna textúra. Trieda, ktorá reprezentuje potravu, má navyše vlastnosť energy. Vďaka tejto vlastnosti sa potrave zmení textúra, vďaka čomu je používateľovi na pohľad jasné, či je potrava celá, alebo už len nejaká jej časť.

Samotný svet je reprezentovaný triedou World. Svet má pevne dané rozmery, ktoré vyjadrujú vlastnosti width a height (počet políčok = width x height). Ďalším atribútom, ktorý je potrebný pre vykreslenie sveta na obrazovku, je aktuálna šírka políčka. Vďaka zmene tohto atribútu sa dá

zabezpečiť približovanie a odďaľovanie sveta. Posledným atribútom je dvojrozmerné pole, v ktorom sú zachytené informácie o každom políčku, konkrétne či sa na ňom nachádza stena, alebo nie. Podľa tohto atribútu sa pre každé políčko správne zvolí textúra.

Jednoduchý diagram tried pre vizualizačnú časť je možné vidieť na obrázku (Obr. 8).



Obr. 8 – Diagram tried vizualizačnej časti

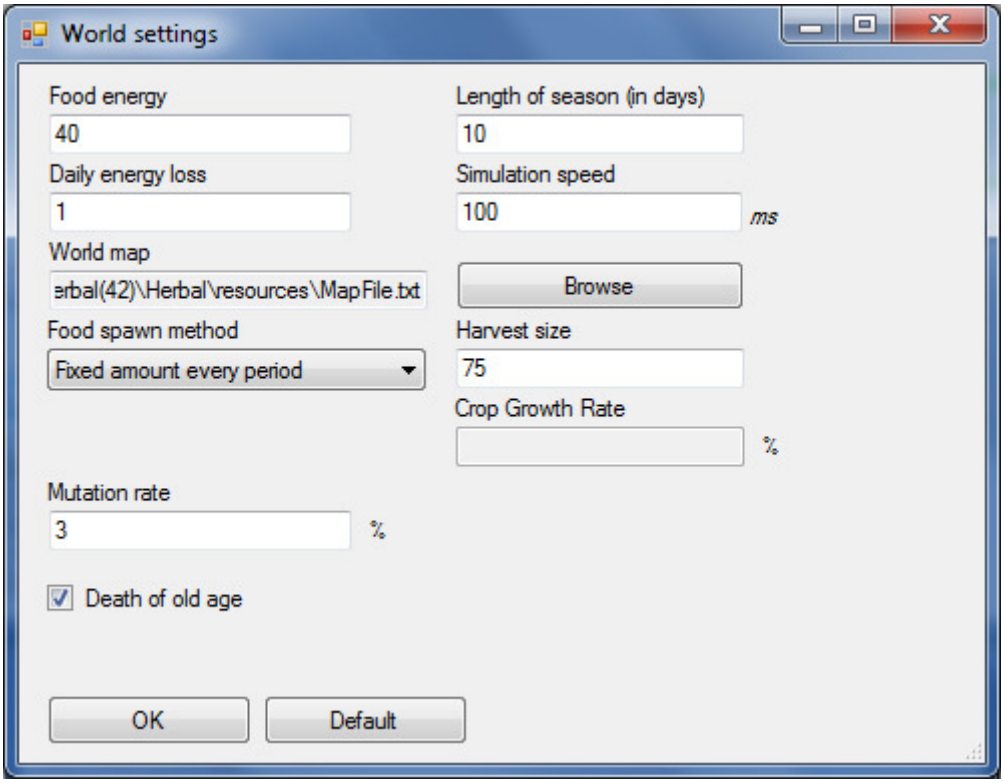
3.2.2 GUI

Aplikácia je navrhnutá ako klasická oknová aplikácia v prostredí Windows, využívajúca pre grafické používateľské rozhranie známe ovládacie prvky z tohto prostredia. Aplikácia pozostáva z 5 formulárov, z toho jeden tvorí vizualizáciu. Okno s vizualizáciou obsahuje tlačidlá, ktoré otvárajú zvyšné formuláre. Zvyšné 4 formuláre slúžia na nastavenie rôznych parametrov aplikácie.

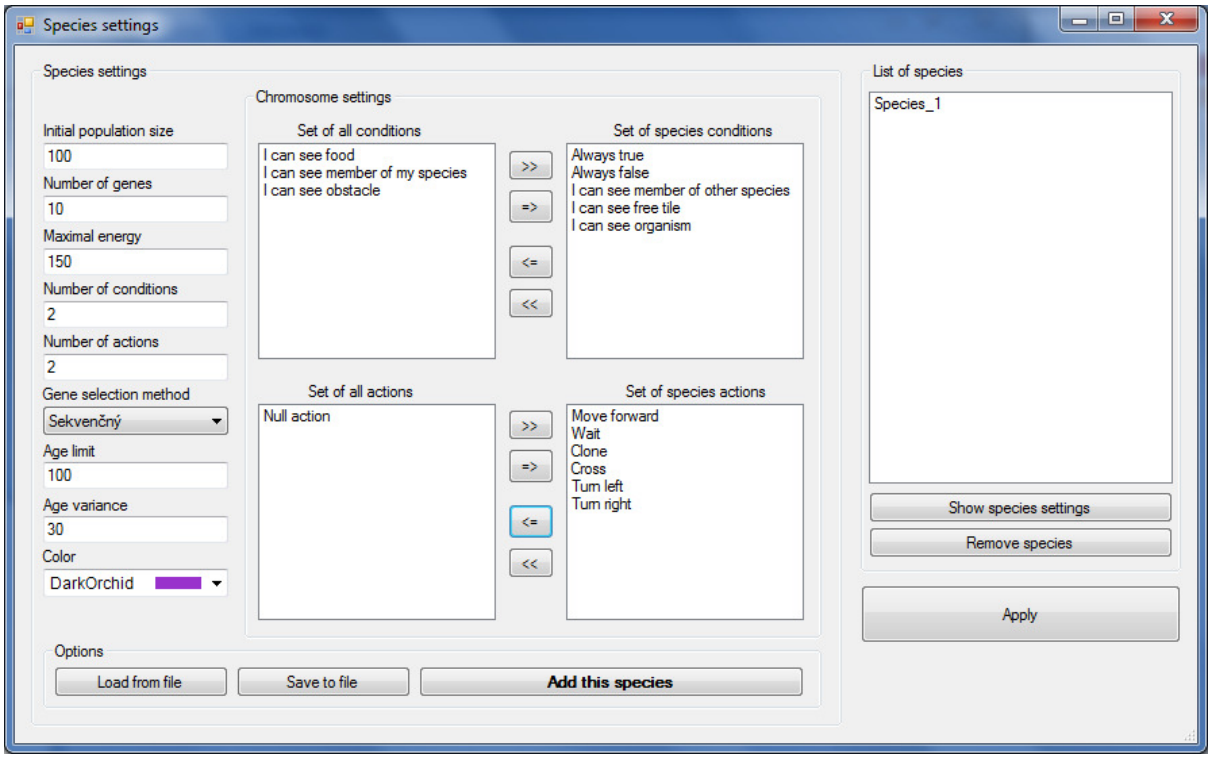
Formulár s názvom World setting (Obr. 9) umožňuje nastaviť parametre simulovaného sveta ako energiu potravy, dĺžku jednej sezóny, hodnotu dennej energie, spôsob pridávania potravy, pravdepodobnosť mutácií, povoliť/zakázať smrť na starobu a ešte slúži na výber súboru s mapou a nastavenie rýchlosti animácie.

Formulár Species setting (Obr. 10) sa využíva na nastavenie simulovaných organizmov. Umožňuje nastaviť rôzne parametre, ktoré je potrebné definovať každému druhu ako počet organizmov daného druhu pri začatí simulácie, maximálnu energiu, počet génov, počet podmienok a akcií v géne, spôsob výberu génu, maximálny vek dožitia, vekovú odchýlku a farbu. Okrem toho obsahuje zoznam všetkých podmienok a akcií, ktoré je možné pridávať jednotlivým druhom. Na pravej strane je zobrazený zoznam všetkých vytvorených druhov, ktoré sa spustia pri simulácií. Okrem toho tento formulár umožňuje jednotlivé nastavenia uložiť alebo načítať zo súboru.

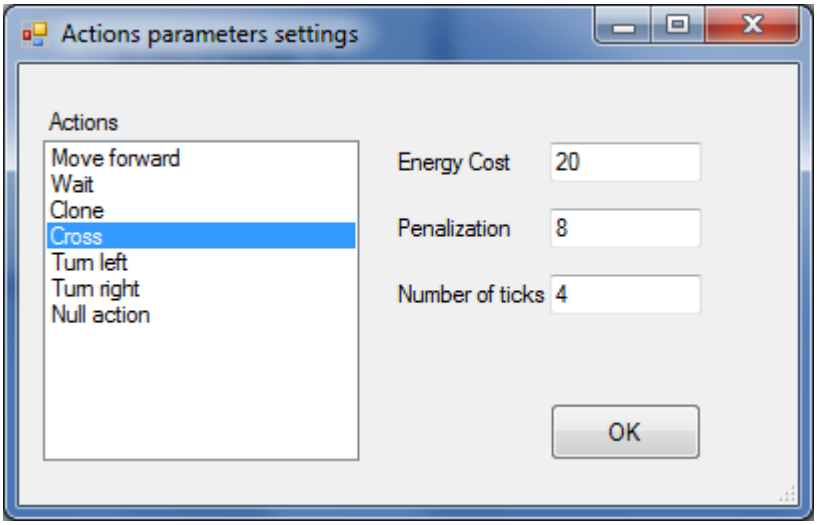
Formulár Actions parameters setting (Obr. 11) je učený na nastavenie hodnôt akcií. Pre každú akciu umožňuje nastaviť hodnotu, ktorá sa odpočíta pri vykonaní akcie (Energy Cost), hodnotu penalizácie (Penalization) pri neúspešnom vykonaní akcie a počet tikov (Numbers of ticks), v ktorých sa akcia vykonáva.



Obr. 9 - Formulár pre nastavenie sveta

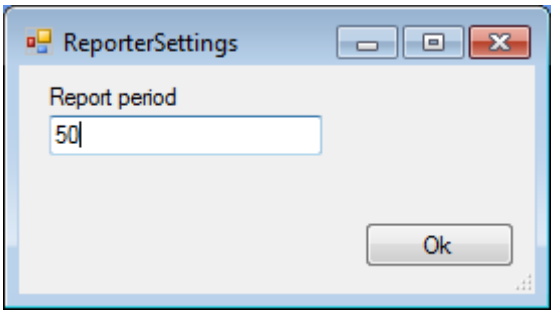


Obr. 10 - Formulár pre vytváranie druhov



Obr. 11 - Formulár pre nastavenie parametrov akcií

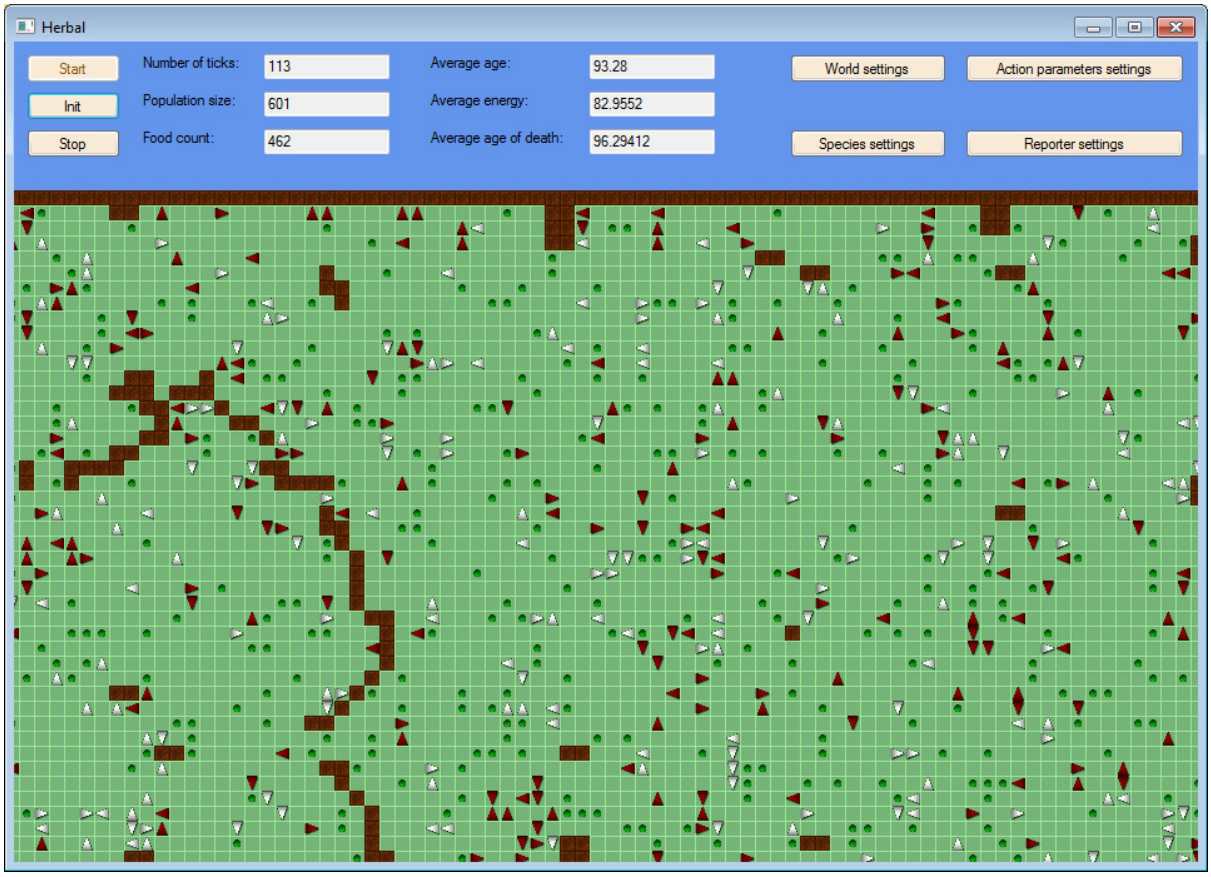
Formulár Reporter setting (Obr. 12) sa využíva pre nastavenie vytvárania reportov. V súčasnosti umožňuje nastaviť iba hodnotu, ako často sa má vytvoriť report, no v budúcnosti sa plánuje tento formulár doplniť o ďalšie možnosti ako nastavenie súboru, do ktorého sa budú vytvárať reporty alebo rôzne spôsoby filtrovania report a iné.



Obr. 12 - Formulár pre nastavenie reportéra

3.2.3 Vizualizácia sveta

Vizualizačné okno sa zobrazí pri zapnutí aplikácie. Slúži ako hlavný formulár celej aplikácie, obsahuje tlačidlá, ktoré zobrazia formuláre spomínané v predošlej kapitole. Na spustenie vizualizáciu sveta je potrebné nastaviť parametre, ktoré boli popísane vyššie. Na obrázku (Obr. 13) je ukážka vizualizácie. Z obrázka je vidieť, že každé políčko má svoju textúru, ktorá slúži na odlíšenie prekážky (steny) od voľného priestoru. Na voľnom políčku sa môže nachádzať herb alebo potrava. Rôzne druhy herbov je od seba možné odlíšiť na základe ich farby (rôznu farbu je potrebné nastaviť pri vytváraní druhu). Pohľad na svet je možné priblížiť alebo oddialiť a okrem toho sa môžeme v rámci sveta s kamerou pohybovať do strán (hore, dole, doľava, doprava). Vizualizáciu je možné kedykoľvek zastaviť a znova spustiť. Okrem spomínaných možností, nám okno vizualizácie poskytuje aj aktuálne informácie o stave sveta ako veľkosť populácie, priemerný vek, priemerný vek dožitia, priemerné množstvo energie, množstvo potravy a informáciu o dĺžke simulácie.



Obr. 13 - Okno so spustenou vizualizáciou

Použité textúry

- Voľné políčko 
- Prekážka 
- Potrava 
- Herb 

4 Návrh testovania

V tejto kapitole uvádzame návrh a opis testovania, ktoré budeme vykonávať spolu s hypotézami, ktoré sme vytvorili a ktoré sa pokúsime overiť.

4.1 Hypotézy

Ako už bolo vyššie naznačené, jedným z hlavných výstupov aplikácie budú zaznamenané štatistické údaje zo simulácií, ktoré umožnia štúdium vplyvu rôznych faktorov (prítomnosť/neprítomnosť istých génov, počet génov, ...) na úspešnosť populácií. Aby bolo možné výsledky študovať a vyvodiť z nich závery bude nutné vykonať veľké množstvo simulácií, avšak my sme sa rozhodli aj bez informácií z týchto simulácií vytvoriť vlastné hypotézy o tom, ktoré faktory a akým vplyvom ovplyvnia populácie najviac. Tu uvádzame naše hypotézy; tieto hypotézy vychádzajú z predpokladu, že podmienky simulácie sú totožné so základnými podmienkami uvedenými v základnej verzii špecifikácie požiadaviek v kapitole 2. Keďže táto špecifikácia je relatívne jednoduchá, je len ťažko možné vytvoriť veľmi odlišné hypotézy. Preto sa v našich hypotézach skôr sústreďujeme na to, ktorý faktor/faktory budú mať najzásadnejší vplyv na istý ukazovateľ populácie (napríklad priemerný vek alebo veľkosť populácie).

V hypotézach sa odvolávame na konkrétne podmienky a akcie, ktoré sú definované v návrhu systému v kapitole 3.1.3 „Návrh konkrétnych podmienok a akcií“.

Úsporný prieskumník:

V tejto hypotéze sa sústredíme na to, ktoré faktory umožnia populácií dosiahnuť čo najvyšší možný priemerný vek. Predpokladáme, že populácie dožívajúce sa priemerne vyššieho veku sa budú na rozdiel od ostatných populácií vyznačovať týmito znakmi:

- vyrovnaný počet génov pre otočenie doprava a doľava; zabráni to krúženiu jedinca v malej oblasti a umožní mu nájsť viac potravy
- malá priemerná cena akcií v jednotkách energie v rámci DNA, t.j.: malý počet génov obsahujúcich energeticky náročné operácie ako napríklad *Clone*, *Cross*, ...
- v prípade výskytu podmienky *SeeFriend* bude akcia otočenie (do ľubovoľnej strany) – ak je pred jedincom iný jedinec, je dobrá šanca na to, že tam už nebude potrava, preto sa treba snažiť zmeniť smer pohybu jedinca

Šetrič energie:

Predpokladáme, že herbovia budú úspešnejší ak budú dosahovať väčší priemerný vek. To sa však nemôže vťahovať na celú populáciu.

- novým jedincom bude postupne miznúť podmienka *AlwaysTrue* a pribúdať akcia *Wait*
- druh s väčšou začiatočnou rozmanitosťou akcií bude úspešnejší

Tuční herbovia:

Predpokladáme, že keď herbovia dostanú priveľa potravy postupom času zlenivejú a takmer stratia schopnosť pohybovať sa.

- Čím viac jedincov v populácií, tým nižší priemerný vek, naopak, čím menej jedincov v populácií, tým vyšší priemerný vek
- Čím bude viac potravy, tým menej sa bude vyskytovať akcia pohybu a naopak čím menej potravy, tým viac akcií pohybu a otáčania

Pud sebazáchovy:

Predpokladáme, že pri dosiahnutí ustáleného stavu organizmov bude možné určiť stratégiu prežitia

- vzniknuté organizmy budú buď aktívne (snažia sa prejsť čo najväčšiu časť mapy) alebo pasívne (šetria energiu – hýbu sa len v nutných prípadoch)
- schopnosť týchto organizmov prežiť bude približne rovnaká

Špecializácia videnia:

Táto teória sa zaoberá senzormi a týka sa sveta, v ktorom budú zberači aj lovci.

- Zberači budú mať väčšiu schopnosť prežiť ak budú vidieť všetky políčka okolo seba (čo najväčší zorný uhol)
- Lovci budú mať väčšiu schopnosť prežiť ak budú vidieť viac políčok jedným smerom (ak bude možné viac ako jedno pred seba, inak im bude stačiť políčko pred sebou a 2 diagonálne vedľa nich, ostatne už nebudú mať vplyv na úspešnosť jedinca)

5 Návrh cieľov pre finálny produkt

V tejto kapitole uvádzame návrh cieľov pre finálny produkt. Viaceré z tu uvedených cieľov už boli spomenuté priebežne v kapitole 2 „Špecifikácia požiadaviek“ a tu ich uvádzame pre kompletnosť a prehľadnosť.

Rozšírenie funkcionality aplikácie

Toto rozšírenie je opísané v prípadoch použitia uvedených v podkapitolách 2.1.12 až 2.1.21.

Rozšírenie sady akcií a podmienok

Pre finálny produkt máme za cieľ rozšíriť základnú sadu podmienok a akcií prototypu.

Vylepšenie používateľského rozhrania

Pre finálny produkt máme stanovený cieľ ľahkej a intuitívnej použiteľnosti – preto musíme prepracovať výzor a usporiadanie jednotlivých grafických prvkov rozhrania aplikácie. Jedným z dôležitých bodov týchto úprav bude nutnosť poskytnutia všetkých potrebných informácií používateľovi – ide napríklad o opis jednotlivých parametrov v grafických oknách pomocou tooltipov a podobne.

Optimalizácia aplikácie

Ide najmä o optimalizáciu vizualizácie, ktorá najviac spomaľuje vykonávanie programu. Pri tejto optimalizácií bude nutné najmä zefektívniť spôsob vykresľovania sveta vo vizualizačnom okne, ktorý je momentálne neefektívny, keďže sa napríklad prekresľujú aj tie prvky, ktoré sa od posledného prekreslenia nezmenili a podobne.