VoSSEN EN KONIJNEN

Projectverslag

Hanzehogeschool Groningen

PIV1J

Rick Wolthuis

Tim van Veen

Tim Zijlstra

Rutger Walma

Robin Slijfer

3 februari 2015

# Inhoudsopgave

[Fouten die nog aanwezig zijn in het programma 2](#_Toc410682716)

[Inleiding 3](#_Toc410682717)

[Probleemanalyse en eerste versie simulator 4](#_Toc410682718)

[1.1 Probleemstelling. 4](#_Toc410682719)

[1.2 Analyse basisversie 4](#_Toc410682720)

[1.3 Varkenscyclus 5](#_Toc410682721)

[1.4 Uitbreidingen volgende versie 5](#_Toc410682722)

[Aanvullingen en verbeteringen op eerste versie 6](#_Toc410682723)

[2.1 Gras, jager en wolf 6](#_Toc410682724)

[2.4 Animatiesnelheid slider 6](#_Toc410682725)

[2.5 Views: Histogram, cirkeldiagram en lijndiagram 7](#_Toc410682726)

[2.6 Refactoring 7](#_Toc410682727)

[Reflectie 8](#_Toc410682728)

[Bijlage A – UML Klassediagram 9](#_Toc410682729)

[Bijlage B – Sequencediagram 10](#_Toc410682730)

[Bijlage C – Projectledenlijst 11](#_Toc410682731)

# Fouten die nog aanwezig zijn in het programma

Alle modellen (zoals de vossen, konijnen, wolven, etc.) zijn nog niet helemaal gebalanceerd, wat kan resulteren in een totale overname van één soort model.

De jagers blijven niet op een stabiel aantal in de simulatie, de bedoeling achter de jager is dat deze nooit zou doodgaan of nakomelingen zal maken.

# Inleiding

Dit rapport is geschreven naar aanleiding van het maken van een simulator genaamd ‘Vossen en Konijnen’. Deze simulator kan een voorspelling maken over hoe de populatie van vossen en konijnen zich ontwikkelt in een natuurgebied. De simulator is zo gebouwd dat de uitkomsten altijd anders gegenereerd worden. Door gebruik te maken van deze simulator kunnen trends in de ontwikkeling van de vossen en konijnen herkend worden, die mogelijk in de realiteit van pas kunnen komen.

De simulator bestaat uit een raster van vierkantjes dat het gebied moet voorstellen. In dat raster zijn sommige blokjes blauw gekleurd en sommige geel. De blauwe blokjes staan voor vossen, de gele voor konijnen. De vossen zijn zo geprogrammeerd dat ze zich voortplanten en konijnen binnen een bepaalde afstand opeten, de konijnen planten zich sneller voort. Zo ontstaat er een constante strijd tussen vossen en konijnen in het gebied.

In de komende hoofdstukken wordt beschreven wat het doel is van het project en welke verbeteringen er zijn gemaakt in de simulator ten opzichte van de vorige versie. Ook wordt er beschreven wat mogelijke uitbreidingen zijn voor een volgende versie.

Eerst wordt behandeld hoe de eerste versie van de simulator tot stand is gekomen en wat de problemen ermee waren. Daarnaast valt er te vinden welke tests allemaal zijn uitgevoerd om de fouten in het programma te vinden. Daarna zijn alle verbeteringen ten opzichte van de eerste versie te vinden. Ook wordt er gekeken of de geplande uitbreidingen gelukt zijn.

# Probleemanalyse en eerste versie simulator

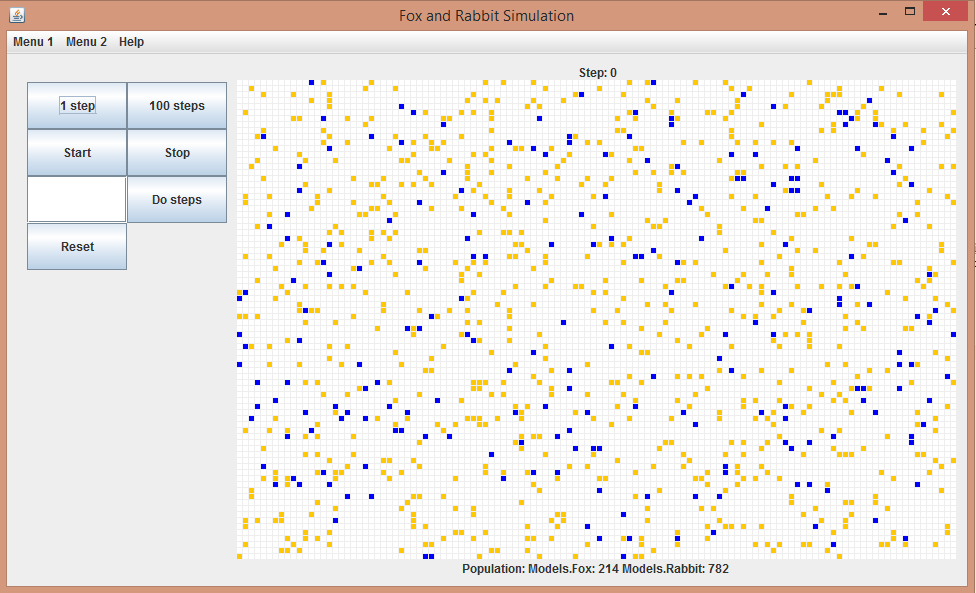
## 1.1 Probleemstelling.

Het roofdier en de prooi hebben een nauw verband met elkaar. Ze hebben elkaar nodig om de populatie gebalanceerd te houden. Deze balans is erg belangrijk voor het behouden van een goede cyclus waarin beide soorten kunnen leven en het gebied waarin ze leven niet beschadigd wordt.

Prooi en roofdier kunnen dus eigenlijk niet zonder elkaar, al zou de prooi dat zoo graag willen. Zonder prooi sterven de roofdieren al gauw uit door gebrek aan voedsel. Zonder roofdieren plant de prooi zich zo gemakkelijk voort dat het leidt tot over populatie en voedseltekorten. De over populatie zorgt dan weer voor schade voor het gebied. Het is dus erg belangrijk om een goede balans te houden tussen de twee zodat het gebied niet overheerst wordt door de prooi of het roofdier.

## 1.2 Analyse basisversie

De opdrachtgever heeft behoefte aan een programma dat goed simuleert welke situaties er kunnen voorkomen met vossen en konijnen. Omdat het programma elke keer een willekeurig resultaat geeft kan de opdrachtgever via het programma vaststellen wat de meest voorkomende eindresultaten zijn. Het programma moet in staat zijn korte en langere periodes te simuleren, zodat de opdrachtgever kan zien wat de ontwikkelingen op korte en lange termijn zijn.

Als het programma gestart wordt heeft de gebruiker de mogelijkheid een hoogte en breedte van het raster aan te geven. Als er geen waarden worden ingevoerd worden de default waarden van 140 bij 100 ingesteld. Daarnaast beschikt het programma over een ‘Stap 1’ en een ‘Stap 100’ functie waar elk dier respectievelijk 1 of 100 stappen zet. Ook is er een ‘Do Steps’ functie, hiermee kan de gebruiker zelf het aantal stappen invoeren en die simuleren.

*De eerste versie van de simulator met enkele basis functionaliteiten*

Als eerste is het programma getest bij een hoogte en breedte van 50 bij 50. De trend die waarneembaar is, is dat het altijd uitdraait op dat of alle dieren uitsterven of de konijnen overheersen. De overheersing door de konijnen komt vaker voor; zo’n 8 van de 10 keer dat er gesimuleerd wordt, al blijft dit natuurlijk totaal willekeurig. Dit gebeurt meestal binnen de eerste 2000 stappen. Als de vossen de konijnen uitroeien en daarna zelf uitsterven, gebeurt dit meestal al binnen de eerste 1000 stappen. Er zijn uitschieters naar meer dan 5000 stappen waar het systeem in evenwicht blijft, maar het gebeurt vaker dat de konijnen al snel de overmacht hebben. Als de vossen winnen, komt dit doordat de vossen de konijnen in een hoek drijven, en omdat het raster niet heel groot is gebeurt dit snel.

Bij een raster van 140 bij 100 blijft het systeem heel lang in balans, dit komt doordat de konijnen meer ruimte hebben en dus minder snel ingesloten worden door de vossen. Het programma heeft bij verschillende tests ruim de 10000 stappen gehaald zonder dat er een soort overheerste. Door deze constateringen zijn de default waarden voor het raster 140 breed en 100 hoog.

## 1.3 Varkenscyclus

De balans die te zien is in de vossen en konijnen simulatie komt aardig overeen met de varkenscyclus/marktwerking. De essentie van de varkenscyclus is dat als ergens een overschot van is, de prijs daarvan daalt en de vraag daarmee toeneemt. Doordat de vraag toeneemt, verdwijnt het overschot en stijgen de prijzen weer. Dit gaat zo door totdat er weer zo veel geproduceerd wordt dat er opnieuw een overschot is en de prijs weer daalt.

In de simulatie valt iets dergelijks te zien. Als er een overschot aan konijnen is reageren de vossen daarop door ze op te eten, daardoor komen er minder konijnen. Doordat er minder konijnen zijn, sterven de vossen uit door gebrek aan voedsel en neemt de ‘vraag’ naar konijnen af. De konijnen hebben dan weer genoeg tijd om zich weer voort te planten waardoor er weer een overschot ontstaat.

## 1.4 Uitbreidingen volgende versie

Het project gaat nog uitgebreid worden met een derde object, namelijk gras. Gras wordt alleen gegeten door konijnen, en voegt een extra factor aan die simulatie toe die belangrijk is voor het overleven van de konijnen. Daarnaast worden er jagers toegevoegd. Jagers jagen op zowel vossen als konijnen, ze schieten elkaar niet neer.

# Aanvullingen en verbeteringen op eerste versie

## 2.1 Gras, jager en wolf

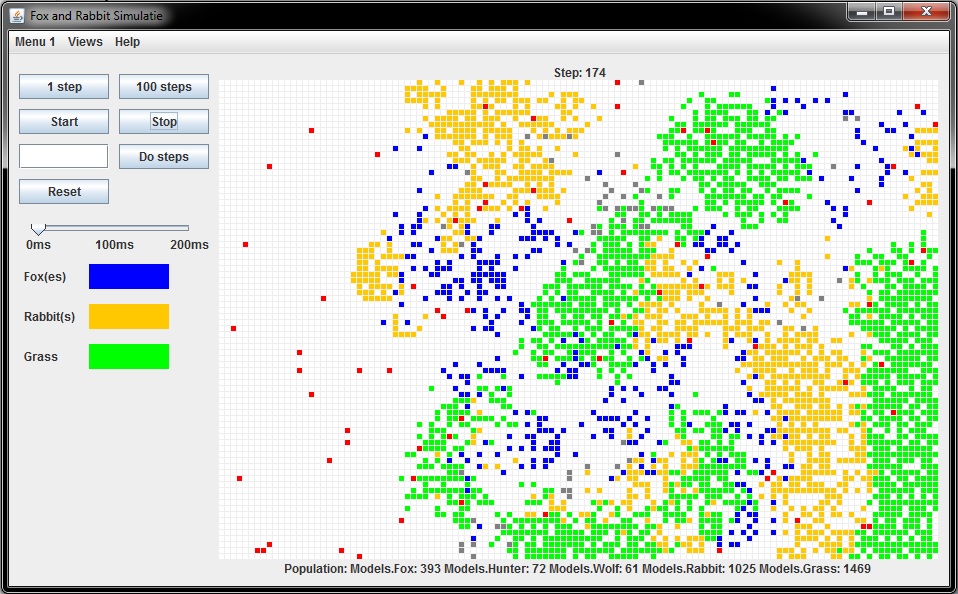
Allereerst is er gras aan de simulatie toegevoegd. Zoals ik het vorige hoofdstuk al beschreven, wordt het gras alleen gegeten door konijnen. De konijnen hebben gras nodig om niet te sterven van de honger, net als vossen konijnen nodig hebben om niet hetzelfde te doen. Gras plant zich net als vossen en konijnen voort. De kans dat ze zich voortplanten is met 0,15 hoger dan de kans van vossen (0,08) en konijnen (0,10). Deze verhoogde kans is ook nodig omdat gras bij lange na niet zo lang leeft als een vos of een konijn. Daarnaast zijn er voor de duidelijkheid kleurlabels toegevoegd die aangeven welke kleur bij welk object hoort.

De jager is ook toegevoegd aan de simulatie. De jager is zo ingesteld dat hij een aantal dieren binnen zijn directe omgeving afschiet. De jager jaagt uiteraard op alle dieren maar niet op andere jagers, de jager kan zelf niet aangevallen worden door dieren.

Ten slotte zijn er wolven toegevoegd. De wolven jagen op vossen en konijnen maar worden gejaagd door de jagers. De wolf gedraagt zich net als een vos, alleen eet hij beide dieren.

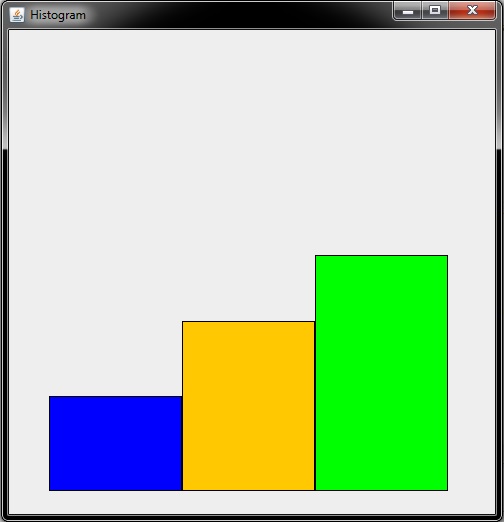
## 2.4 Animatiesnelheid slider

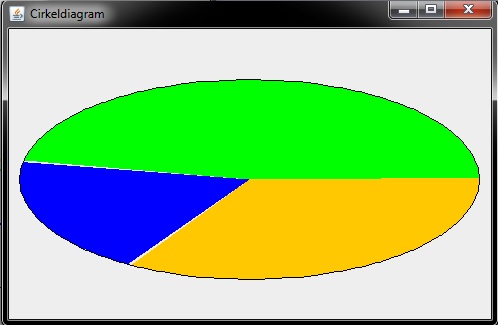
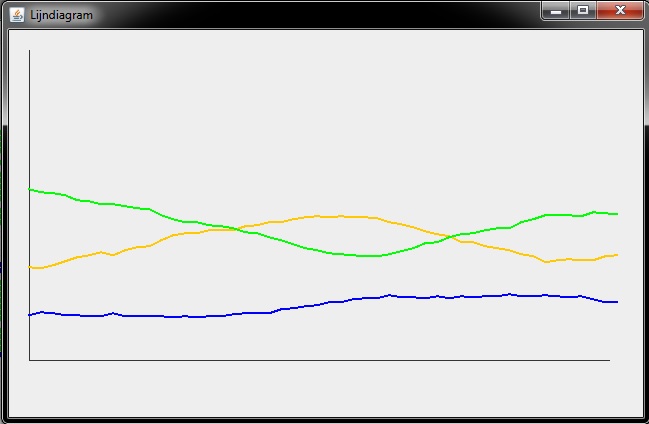
Een andere toevoeging is de slider die de snelheid van de simulatie verandert. Met behulp van de slider kan de animatie teruggebracht worden tot een snelheid van 200ms. De standaard animatiesnelheid is te hoog om echt te zien welk konijn welk gras opeet of welke vos welk konijn opeet. Door de simulatie langzamer te laten lopen, kan er beter gezien worden wat er nou precies gebeurt.

*Versie 2 van de simulatie*

## 2.5 Views: Histogram, cirkeldiagram en lijndiagram

Ook zijn er verschillende views toegevoegd aan de simulatie, die via een drop down menu bereikt kunnen worden. In het menu ‘Views’ krijgt de gebruiker de keuze uit een histogram, cirkeldiagram en een lijndiagram. Deze diagrammen worden geopend in een los window, en kunnen dus ook alle drie tegelijkertijd met de simulatie gedraaid worden. De diagrammen geven de verhouding tussen de vossen, konijnen en gras aan.





## 2.6 Refactoring

De code is meer begrijpelijker en leesbaarder geworden. Met behulp van comments kunnen ook mensen die niet veel ervaring hebben met programmeren begrijpen wat waar gebeurt. Ook is commenting erg handig om aan andere programmeurs in het project te laten weten wat er waar gebeurt.

Daarnaast is de code leesbaarder gemaakt door het effectiever gebruik maken van witregels tussen verschillende delen code. Het inspringen van regels is ook een manier die gebruikt is om de code leesbaarder te maken en duidelijker te maken wat bij wat hoort.

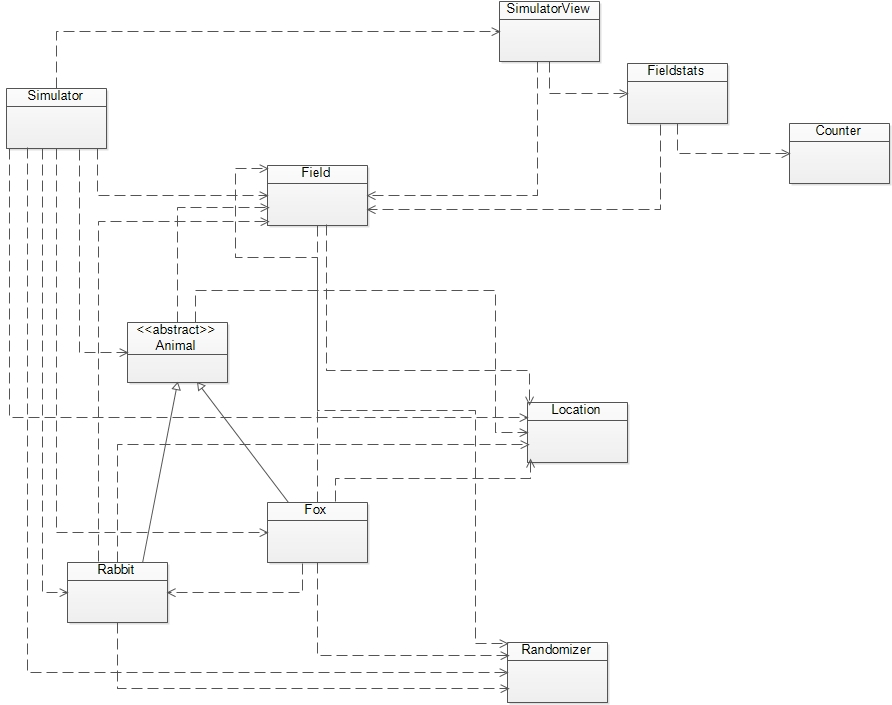
# Reflectie

De verbeterde versie van de simulator is een hele stap vooruit in vergelijking met de basis versie. De simulator is uitgebreid/verbeterd op het gebied van zowel code als gebruiksvriendelijkheid. Vooral de uitbreidingsmogelijkheden zijn er ver op vooruit gegaan. Het programma is daardoor minder beperkt geworden en is in de toekomst nog eenvoudiger uit te breiden.

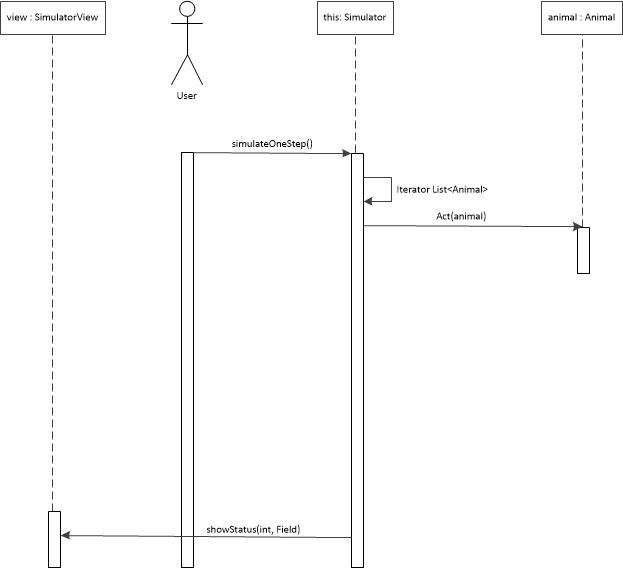
Het opgeleverde programma beschikt over een gebruiksvriendelijke GUI en code met begrijpelijk commentaar en javadoc. We zijn tevreden over de gehaalde functionaliteiten. Helaas waren niet alle uitbreidingen haalbaar en die zijn niet toegevoegd tot deze versie van het programma maar zullen eventueel in de toekomst gerealiseerd kunnen worden. Deze uitbreidingen betrekken zich voornamelijk richting visuele functies en hebben verder geen invloed op de loop van de simulatie. Een uitbreiding die eigenlijk wel hoog op het lijstje stond was een ziekte onder de dieren, maar dit bleek niet haalbaar.

Al met al is duidelijk dat het programma werkt en voldoet aan de functionaliteit van wat er van het programma verwacht wordt. Daarmee is het programma gebruiksklaar voor de daarvoor bestemde doeleinden

# Bijlage A – UML Klassediagram



# Bijlage B – Sequencediagram



# Bijlage C – Projectledenlijst

**Rick Wolthuis  
Taken:**

* **Programmeren van applicatie**
* **Ondersteuning bij documentatie**

**Tim Zijlstra  
Taken:**

* **Programmeren van applicatie**
* **Klassen diagram, Sequence diagram**
* **Ondersteuning documentatie**

**Tim van Veen  
Taken:**

* **Programmeren van applicatie**
* **JUnit, refactoring**
* **Ondersteuning documentatie**

**Rutger Walma  
Taken:**

* **Probleembeschrijving en analyse basisversie**
* **Projectverslag**
* **Documentatie**
* **Ondersteuning bij programmeren van applicatie**

**Robin Slijfer  
Taken:**

* **Documentatie**
* **Voorbereidingen presentatie**