research report rev 12

PROJECT APPLICATION DEVELOPMENT - Inholland University of Applied Science

Abstract

This research report explores the development of an application that results from a possible solution, to a problem taken from an existing ecological system, by using a mathematical model to show the predicted effects of the proposed measure on the ecosystem.

Research by:

572481 - Adu, Stephen

570027- Andreicha, Semida

563453 - Buaron, Tal

573967 - Cholodov, Andrej

566536 - Domela Nieuwenhuis, Jens

With the assistance of Harald Drillenburg and Koos van Tubergen

Table of Contents

[1. Introduction 2](#_Toc471544897)

[1.1. Context 2](#_Toc471544898)

[1.2. Purpose 2](#_Toc471544899)

[2. Central research question and sub-questions 2](#_Toc471544900)

[2.1. Research question 2](#_Toc471544901)

[2.2. Research sub-questions 3](#_Toc471544902)

[3. Research Methodology 3](#_Toc471544903)

[4. Results 6](#_Toc471544904)

[5. Conclusion 6](#_Toc471544905)

[6. Recommendations 6](#_Toc471544906)

[7. Bibliographies 6](#_Toc471544907)

[8. Appendices 6](#_Toc471544908)

# Introduction

Mathematical engineering is based on developing application models for making predictions and find desirable solutions to real-life problematic situations. While working towards the development of an application, one comes across several challenges which fluctuates the approach of development. Challenges can be encountered when determining the model of the application, due to the observation and collection of data which is relevant to use as variables in a mathematical formula. Thus, parameters are going to be divided into categories such as the essential and trivial factors, then later implemented to improve the model. Therefore, due to the level of complexity that the model may hold it is mandatory to not only implement an application which gives as an output statistical numbers but visuals as well, such as graphs.

# Context

Oostvaardersplassen, a persevered ecological system, which has many animals living there including the main three large herbivores: wild horses, wild cattle, and deer; and other foremost species such as geese and birds of prey. Conjecting that no major predators are present in the ecological system and the herbivores prohibited migration, unbalance has been settled in the Oostvaardersplassen preservation.

Complicating the situation is the fact that there are a lot of geese, particularly during winter, and they consume the same type of food as the large herbivores do. The competition and interact between the herbivores and the geese for food can be quite severe, leading some to extreme situations such as death.

# Purpose

The purpose of this project is to work towards a possible solution of a problem taken from an existing ecological system, Oostvaardersplassen. Based on a mathematical model for the system and an application doing the number crunching to graphically illustrate the predicted effects of the proposed measures on the ecosystem.

# Central research question and sub-questions

# Research question

Main Research Question of the project:

What will happen to the populations of deer, cattle, horse and geese if releasing a number of foxes in the preserve (with the number of foxes being the free variable)?

Main Research Question for this research paper:

How do the herbivores and the geese compete for the grass in an enclosed area?

# Research sub-questions

Sub-Research Questions for this research paper:

* + 1. What is the rate of grass consumption of each animal?
    2. How do animal populations change as a result of grass availability?
    3. What are the populations of each animal?
    4. What is the start data and what is known from historical data?
    5. Which existing mathematical models can fit to the problem?
    6. How can the application hold all the implementations what are being asked?

# Research Methodology

The methodology explored in this project reflects the competition for food, taking place between herbivores and geese, in an enclosed ecosystem, Oostvaardersplassen. By means of collecting data and modeling equation based on the data, the research will then attempt to provide an accurate prediction based on fluctuating data. The **Data collection** is based on the information found on the Website of Oostvaardersplassen1.1 (Startpagina Staatsbosbeheer. Het zit in onze natuur. n.d.), which is related to the numbers of herbivores and geese during different seasons, the amount of grass they eat, death rate and any other related data.

The **Equation research** is mostly focused on the equations of the grass growth. The research conducted to find equations is based on literature2.2 (Gotelli, N. J. 2008). Transforming grass amount into actual weight so comparison can be made to the amount of food the herbivores and geese are eating.

Once the data collection process and equation research will be finalized, the creation of the application model will begin. By creating class models of UML [[1]](#footnote-1)will help to visualize how the application will look like and what it should do. As soon as, the UML model for the application will be completed the focus then turns to the back-end work, implemented using Java. After having the engine and the whole map of the Oostvaardersplassen in program, the application will then display in a GUI[[2]](#footnote-2), by JavaFX[[3]](#footnote-3).

During the project development several limitations may occur, that could harm the final result of the application. Some of those **Limitations** for this research are the time frame which project needs to be finished. The information availability constraints which might be lacking data or not accurate at all. Another challenge is the interdependence on the other group that is investigating relating variable. This research is limited scope to Oostvaardersplassen.

The **first sub-question** is getting the information about the *grass consumption* of each animal. The way in which the answer to this question will be given, is by researching about the amount of grass each of the animals in Oostvaardersplassen needs in order to continue living. There is a lot of information about herbivores and geese grass consumption on the internet, however only the most similar specie of each animal of Oostvaardersplassen will be chosen, then take the figures found and implement it in the application.

In concordance to the website3.3 (Intake requirements n.d.), in regards to the data collection concerning with the food consumption of the *Red Deer*, it shows a distribution of a 50/50 rate of male/female dears. By means of data shown in a graph, it was deducted that the average deer weights about 110kg, and consumes about 2.5kg of Dry Matter each day.

According to the website*4.4* (Konik 2016), used to collect data about *Horse* food consumption, the average horse consume every day around 2.5% of DM[[4]](#footnote-4); depending on the horse‘s body weight. According to the source, the horses in Oostvaardersplassen weigh between 350kg-400kg, it was decided on taking an average body weight of a horse of 375kg, which consume 9.4kg of DM each day.

From the source5.5 (Cattle 2017), used to generate information about the food consumption of a *Cow*, can be deduced that the average cow consume 2.5% of its body weight in DM each day. The average weight of a cow in Oostvaardersplassen is 600kg, which means that each cow consume 15kg of DM every day.

Due to lacking information in the data collection in regards to the food consumption of a “wild *Goose*”, a farm goose was decided to be taken as source for indication. According to the sourc*6.6*, the average goose eats everyday around 0.15kg of DM.

In order to answer the **second sub-question**, research needs to be done about what is the formula that most fit the situation, in order to know how the availability of grass would affect the animals that are there. This information exists in the ecology book that was given as a reference.

The way in which the **third sub-question** will be answered, is by getting the existing numbers of Oostvaardersplassen herbivores and geese from the given information that was received and the information that is on Oostvaardersplassen website. Afterwards the process of implementation of those figures in the application will begin, so that the outcome would be as accurate as it can be.

In order to show the **fourth sub-question**, historical data of Oostvaardersplassen need to be accessed. There is a given website that can give all that information in graphs and actual numbers. The starting data would be the latest figures that are available in 2016, this would be the actual information that is going to be used as the starting data.

In the **fifth sub-question**, figures what mathematical models would fit the problem, deduced from the received ecology book which contains all the formulas that can be related to the problem. Research about that book and figure needs to be conducted in order to know which of the existing formulas will help with the calculation, in the best way and most accurate.

In order to implement all the information that is collected, for the **sixth sub-question** the use of both front end development and backend development, with Java and JavaFX. With the intention of showing graphs, which outcome is based on the input that will be inputted by the user.

In view of the fact that the application is built dynamically in the *front end development*, the best choice for creating the Graphical User Interface (GUI) was by utilizing Scene Builder[[5]](#footnote-5). Scene Builder was a shaped fit, in regards to the application’s needs, not only because of the affluent maintainability but also due to its support; as well as the trivial factor of previous work experience with this tool.

The GUI was designed with ease of use in mind, to be simple yet effective. It consists of Menu Bar, Text Fields, Buttons, Radio Buttons and Graph, which visually represent the data.

Inside the *Menu Bar* three main items can be found. A primary item is the close functionality, which allows the user to exit the application, and is found under “File >Close”. Another item is located in “About us”, which opens a window with a brief introduction about the application creators.

“Cattle”, “Red Deer”, “Horses”, “Geese” and “Years” are the *Text Fields* which prompt the user to enter Custom Data, numerical values specifically, that will be used in the Prediction model. The fields are checked by application for correct entry; if user enters a word value instead of a numerical one, or leaves area empty – an error message will be displayed.

The predominant *buttons*, for the application, are the ones which influence the most the outcome of a prediction. “Predict” is one of the buttons, and its task to take entered data and sends it to Model to execute calculation, then display results in View. A second button is “Show Historical Data”, which draws the chart of historical data for cattle, horses and deer from 1982 to 2015. The next button resets the Text Fields and Graph view, so that the user can reenter data for a new prediction, entitled “Clear Data”. The last button, “Grass Left”, displays a message in console about the grass availability at the end of the calculation.

The main purpose of the *Radio Buttons* is to manipulate chart view. The user can select to display one or more preferred animal graphs, or use “Totals” button to show all graphs together.

Since the application displays its results visually, by drawing charts which then will be compared and analyzed by user, the *Graph View* takes up the largest part of application window. The chart will automatically adjust its scale to given timeframe and graphs. Each animal is assigned unique color for better readability.

In the *backend work*, the development of the software side, the language the code was written in is Java. By using classes and methods, which are referred to by name and invoked at any point in a program to get a functionality. Below can be found a brief description of the main methods that make the application functional, which have a more direct association /involvement towards the frontend work.

The core controller behind the *buttons* functionality found on the GUI, for this application, are the predominant methods. One of the methods, is *handleCompute,* which makes the prediction by taking all the text fields as input and puts them inside the formula. By way of using a for loop to predict each year, then showing the output on the graph. A second significant method is *handleHistoricalData*, which shows all the historical data of the nature preserve since 1983 up to 2015. Both of these methods have implemented a *disabling* option, which can be pressed to compute/show historical data buttons again after the first prediction has been made. A third considerable method used to control the buttons is *methodhandleClearData*, which clears the data of the prediction or the historical data which was shown in the graph; enables the show prediction/historical data after the clear data button was pressed. *calculateGrassLeft* is a fourth method for button control, which prints to the console the amount of grass left each year; by taking the text fields as input to deduce the amount of available grass each year.

A special type of buttons used in the application are *RadioButtons*, means that only one button at a time can be selected. The three radio buttons used are *handleAppearanceRedDeer*, which shows and removes the line of the dears on the graph, the *handleAppearanceHorses*, showing and removing the line of the horses from the graph, and lastly *handleAppearanceCows*, shows as well as removes from the graph the lines dedicated to cows.

Another aspect which stands out is the *TextFields* section. Which is a method that takes all of the text fields as input then changing them from text field property to a string. Following with a transition from a string to a double, in order to push it into the formula and make the prediction. This process is true for cattle input, horses input, deer input and years input.

The menu functionality has two main methods *handleExit*, found under file with its main task to exit the application, and *handleAbout,* which shows the creators names for this application in an Alert window and is located under the About us tab.

An important method which is conspicuous is the *isInputValid*, checking if all text fields are being filled correctly with inputs when the buttons have been pushed. Otherwise a warning message with information in regards to what field needs adjustments are needed, in order to proceed with the prediction.

# Results

# Conclusion

# Recommendations

# Bibliographies

*1.1* Startpagina Staatsbosbeheer. Het zit in onze natuur. (n.d.). Retrieved November 23, 2016, from <https://www.staatsbosbeheer.nl/>

2.2Gotelli, N. J. (2008). *A Primer of Ecology* (4th ed.). Sunderland, Massachusetts: Sinauer Associates.

3.3 *Red Deer*:

Intake requirements. (n.d.). Retrieved January 06, 2017, from <http://deernz.org.nz/deerhub/feeding/feeding-deer/intake-requirements>

*4.4Horse* Source:

Konik (2016). . In *Wikipedia*. Retrieved from https://en.wikipedia.org/wiki/Konik

SARAH, R. N. (2016). Nutritional requirements of horses - management and nutrition - veterinary manual. Retrieved January 6, 2017, from <http://www.merckvetmanual.com/mvm/management_and_nutrition/nutrition_horses/nutritional_requirements_of_horses.html>

*5.5Cow* Source:

Cattle. (2017). How much grass does a cow eat each day? Retrieved January 6, 2017, from <http://www.answers.com/Q/How_much_grass_does_a_cow_eat_each_day>

*6.6Goose* Source:

Retrieved January 6, 2017, from <http://wildfowl.wwt.org.uk/index.php/wildfowl/article/viewFile/773/773>

# Appendices

1. UML – Unified Modeling Language is a general-purpose, developmental, modeling language in the field of software engineering, which intends to provide a standard way to visualize the design of a system. [↑](#footnote-ref-1)
2. GUI – Graphical User Interface is a type of user interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation. [↑](#footnote-ref-2)
3. JavaFX - software platform for creating and delivering desktop applications, intended as the standard use of GUI library for Java. [↑](#footnote-ref-3)
4. DM – abbreviation for dry matter; [↑](#footnote-ref-4)
5. Scene Builder – is a visual layout tool for JavaFX applications. The JavaFXScene lets a user rapidly project a JavaFX application user interfaces, without coding. [↑](#footnote-ref-5)