**TABLE OF CONTENTS**

1. Introduction…………………………………………………………………………………..

1.1 Scope………………………………………………………………………………………

1.2 Related Documents…………………………………………………………………………….

1.3 External Requirements………………………………………………………………………

1.4 Internal Requirements……………………………………………………………………….

1.5 Design issues…………………………………………………………………………………..

2. Internal Design………………………………………………………………………………

2.1 Data Flows………………………………………………………………………………..

3. Unit Test Strategy………………………………………………………………………

4. Additional Information…………………………………………………………………

5. References………………………………………………………………………………

6. Acronyms………………………………………………………………………….

**1. Introduction**

This software is designed to simulate the life cycle of various animals. Animals survive based on the availability of food i.e. other animals or plants, hence a food chain is simulated.

**1.1 Scope**

The application consists of a grid where animals and plants are simulated. Each animal is classified depending on their diet into a category. In this application we have plants, herbivores and omnivores. Depending on the classification of the animal, the application will decide which animal can consume which plant or other animal provided in the given grid. The animals do not reproduce and are added by the application.

**1.2 Related Documents**

Class Diagrams on page

**1.3 External Requirements**

1. By the given image of Sask Wildlife software can be created
2. Simulate the life cycle of the animals based on their habitat.
3. Seeking for food
4. Animal can die due to hunger or eaten by other animals.
5. Animals can move a couple space by walking on ground or flaying on the sky.
6. Vegetation like plants cannot move but can be eaten by other animal
7. Amount of movement made by each animal is different
8. Animal cannot die naturally either hunger or eaten.
9. Animal dies if it reaches hunger limit. Hunger is decreased when an animal eats.
10. Tree and shrubs don’t re-grow.
11. All animals can die depend on their ages or eaten by other animals, but some animals cannot be eaten ,so these animals has life limit
12. Each animal have different hunger level. The hunger level increase periodically until it reaches the limit
13. Animals can be at the same position at a time.
14. Simulation will move one animal at a time
15. There will be no fight for survival between a prey and predator
16. Some exceptions will be made for animals to allow them to eat other animals near them
17. Animal must be on the same position as its food in order to eat
18. Each animal can be consumed or can consume
19. Animal move and eat if they find right food on the same position

**1.4 Internal Requirements**

1. Create class hierarchy
2. Encapsulate classes
3. Automatically grow, prey and terminate instances of animals based on their interaction with other classes i.e. simulation
4. Display animals and plants on the screen using arrays
5. Create and delete instances of classes when on the same grid point.
6. Animals and Plants are created in compile time only
7. Implement can eat functionality to determine which class can terminate which class

**1.5 Design issues**

1. Aquatic life inclusion unknown
2. Human interference in wildlife unknown
3. Weather and clime change has impact of the animal’s life.

**2. Internal Design**

This software has main class which is Animal Kingdom class. This class runs simulation and iterates. They are two main classes Plants and Herbivores which determine the behavior of animals. Another class Omnivores determines behavior for animals which inherits from Herbivore class. These are abstract classes.

The attributes in the Animal Kingdom are name of the plants and animal which is string. Each animal and plants need an ID number to be compared. Age of the plants and animal the type of the age is integer. Also animals and plants have a life limit. The age increases periodically for all animals and plants per iteration. They terminate when age is greater than their life limit or if they are terminated by a class.

They are two classes connected to Herbivores classes. Herbivores class has Hunger and Speed functions as attributes. Hunger and Speed functions determine their distinct behavior in terms of movement and food consumption. Insects and Omnivore inherit from Herbivores. Insects class that contain one method which is Move function method and there are two classes that inherited from Insects class. They are instance-able classes which are Grasshopper and Caterpillar. Omnivore class have one method which is eat Animal function this method means that there are animals can eat other animal. Birds and Mammals classes are inheriting from the Omnivore class. Birds and Mammals class both have the same Move function methods based on how each animals move.

Animals and Plants are added in the Animal Kingdom manually. In every iteration, animals and plants grow until they reach their life limit or are consumed by other animal or die due to hunger. Hunger is incremented accordingly for every animal instance in each iteration. Iteration also moves an animal a time according to it specified ability in its category. On a move, the animal checks for collision with other object. If true, it checks if it can eat the other object or the other object can eat it. If anyone is true, it terminates the animal being consumed and reduces hunger .If none true, then animals reside on the same position and until they move in next iteration.

**2.1 Data Flows**

Sequence Diagrams

**3. Unit Test Strategy**

Testers will add plants and objects and run simulation to observe and record animal’s interaction with each other. The results will be compared with the food chain diagram supplied by the client.

**4. Additional Information**

**5. References**

**6. Acronyms**