Automata Simulation Design Proposal

Document Author(s): Katelyn Bunker

Date: 09/21/16

Design Rationale

The main objects that should be taken into account for the project are an Animal object and an Ecosystem object. The Ecosystem object keeps the grid ecosystem as a 2D String array as well as the number of steps the ecosystem has gone through. The Animal class is actually an interface that is implemented by three ranked Animal classes, the LowAnimal, the MiddleAnimal and the HighAnimal. These three Animal classes contain the Animals low on the food chain, in the middle of the food chain, and high on the food chain. These Animals will know their individual symbol as a char, name as a String, the number of steps since they've bred as an int and the number of steps since they've eaten if middle or high, or the number of steps they've been alive if they're low, all as an int. This method of organization for the Animals and the Ecosystem will be the cleanest, so that the information of the Animals is all kept by the individual Animal, and the Ecosystem being its own object will be able to keep track of where all the data is on the grid.

The data for the Animals and the Ecosystem is read in from a file. This data file is passed into the EcosystemRecordsIO class as the fileName String parameter. This class contains a method readEcosystemFiles to read the file line by line. For the first N number of lines (N being the first token in the file) the method sends the line to a processAnimal method as a String which then creates the Animal object from the data contained in the file. After the N lines, the data is all for the Ecosystem grid so the lines are passed into the processEcosystem method which then creates a new Ecosystem from the given data. Using this design, all the data in the file gets read appropriately and the necessary objects are created.

The bulk of the program is run through the Simulation class. This is the class that run the program and communicates the most with the GUI. It has a method, printGrid, that gets the Ecosystem grid from the Ecosystem class and prints it to the GUI. There is also the takeStep method that starts the process that the program goes through when a step is taken. The bulk of the step process runs through a separate Step abstract class and LowStep, MiddleStep, and HighStep classes that extend the abstract class. The Simulation class has a pointer that runs through the grid one cell at a time and when it comes across a cell that has an animal in it the subsequent Step class is called.

The Step class and those classes that extend it carry the bulk of the programming to actually take a step on the grid. When the Step class is called the correlating high low or middle Step class is called to run through breading, eating, or moving, or none of the above depending on whether the Animal has taken enough steps to breed and there is an available breeding spot, whether there are any animals in connecting cells that can be eaten, or if there is an empty spot available that can be occupied by the moving Animal.

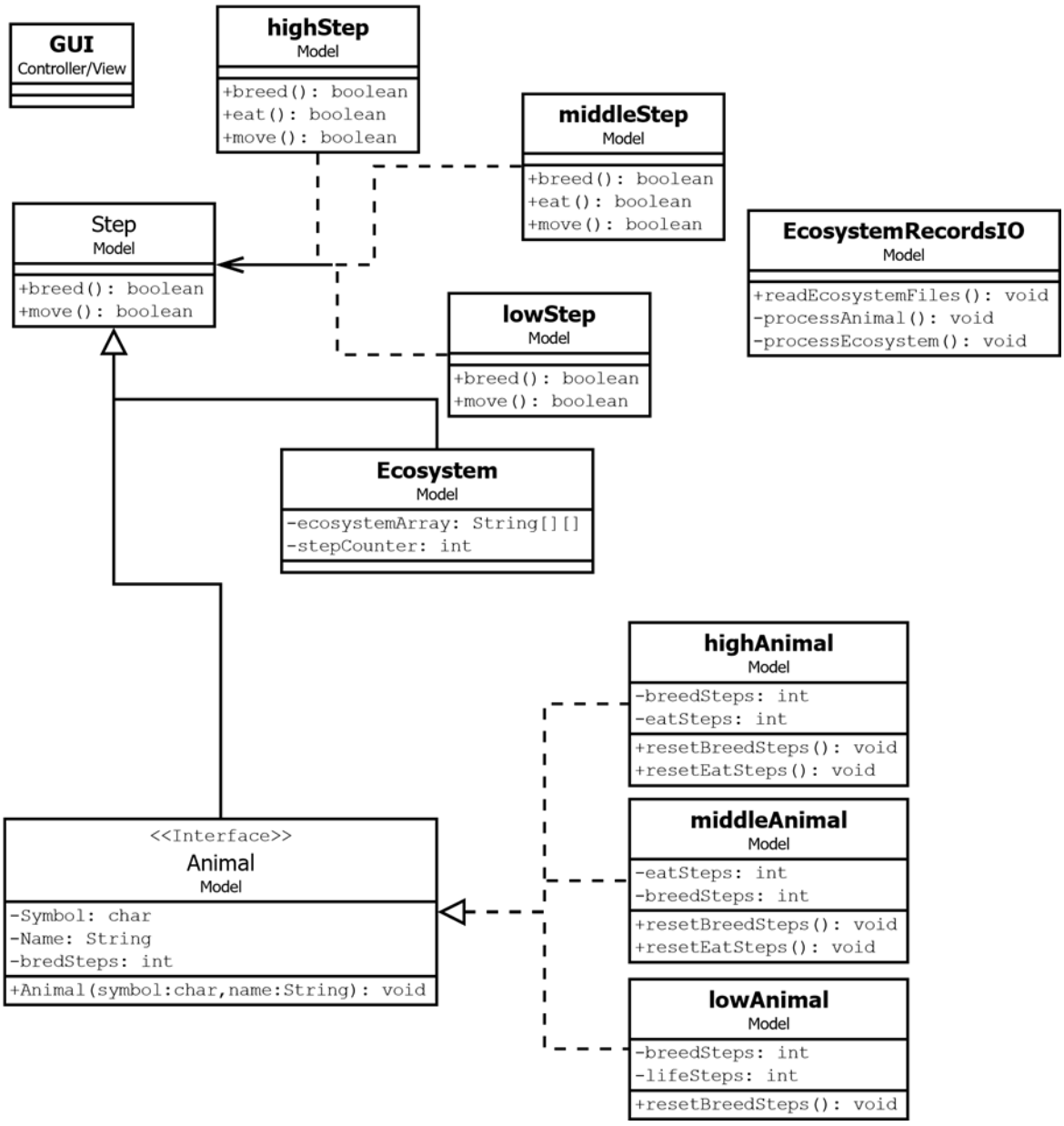

Figure 1: Class Diagram for Automata Simulation

Document Revision History

Date	Author	Change Description
09/21/16	Kbunker	• Started design proposal
09/21/16	Kbunker	• Added the proposal for the Animal, EcosystemRecordsIO, and Simulation classes
09/22/16	Kbunker	• Changed the Animal class to an interface with three classes that implement the interface for the three ranks on the food chain.
		• Added the Step abstract class and the classes that extend it
		• Rewrote most of the document for better flow and to take changes in model into account
		• Added the diagram