CS 673
Assignment #5
Software Requirements Specification (SRS)

Team Snow Crash
Dale Earnest
Jeff Dunn
Dong Luo
Mike McWilliams

Table of Contents

Revision History

Name	Date	Reason For Changes	Version(s)
Dale	27 Oct	Initial Draft	0.1
Dale	31 Oct	Major revisions, added several diagrams	0.2
Dale	02 Nov	Edits throughout, added additional diagrams	0.3
Dale	03 Nov	Added Post Mortem, added definitions	0.4
Dale	03 Nov	Final edits approved	0.5

1.Introduction

The project is an application that allows the user to create or use default critters and see how they might interact in a simulated world.

1.1 Purpose

This document describes the requirements which define the specific functionality of the Evolution Simulation. This document forms the basis of all future detailed design and coding.

1.2 Definitions, Acronyms and Abbreviations

Critters – Instances of a Critter Template that populate the World.

Critter Prototypes – The different base types of critters; there are Plants, Prey, and Predators. The prototype determines how the Critter will interact with the world.

Critter Templates – The model of a Critter that the user may design that contains ranges of Trait values that determine the specific Critters that will populate the world. Each template is defined by a Critter Prototype.

GUI- Graphical User Interface, this describes the user interface based on graphics (icons and pictures and menus) instead of text and uses a mouse as well as a keyboard as an input device.

States – These describe the various states that a critter may be in. A state describes an activity that the critter is doing on the world map and may lead to another state.

Traits – A pair of integer values, assigned by the user, that determine how good a critter is at a given task. This is a measure of how successful the critter will be in the world.

World – A bounded grid upon which critters live, interact, and die.

2.Project Description

2.1Functional Overview

The project consists of a user interface, simulation engine, and file import/export subsystem. The user creates and configures custom critters, starting from default templates, and determines the size of the simulated world. Critters are placed in the world according to the user's specifications when the simulation is started. Following the simulation, the user is presented with the results of the simulation.

During critter-configuration time, the user can allot scores from a points pool to each of various Traits. These Traits determine how the critter interacts with the world and other critters. At any time during this phase, the user may choose to export or import critters to or from files.

When the user is satisfied with his critters, he can run the simulation. The simulation may be paused and resumed, or stopped completely, resetting the turns to zero. The simulation has elements of randomness, so the same inputs may not necessarily generate the same outputs. The user may save the state of the world and its critters during a simulation and open that simulation at a later date.

When the simulation is over, the user is presented with a summary of results. The user may save simulation results and open them at a later date. Analysis of results is left to the user.

2.2 User Interface

2.2.1Overview

There are three screens in the UI: the configuration screen which is the first screen that appears when the user starts the application; the simulation screen which runs the simulation; and the results screen which displays the results of a simulation. Each screen consists of the window frame, menu bar, and central panel.

From the configuration screen, the user can either start or load a saved simulation, and go to the simulation screen, or the user can load saved results and go to the results screen.

From the simulation screen, the user can return to the configuration screen to define a new simulation or stop the simulation and view the results.

From the results screen, the user can return to the configuration screen.

2.2.2Configuration Screen

The configuration screen is what the user sees first upon starting the application. It allows the user to create and edit critters and to design the world upon which the simulation will run.

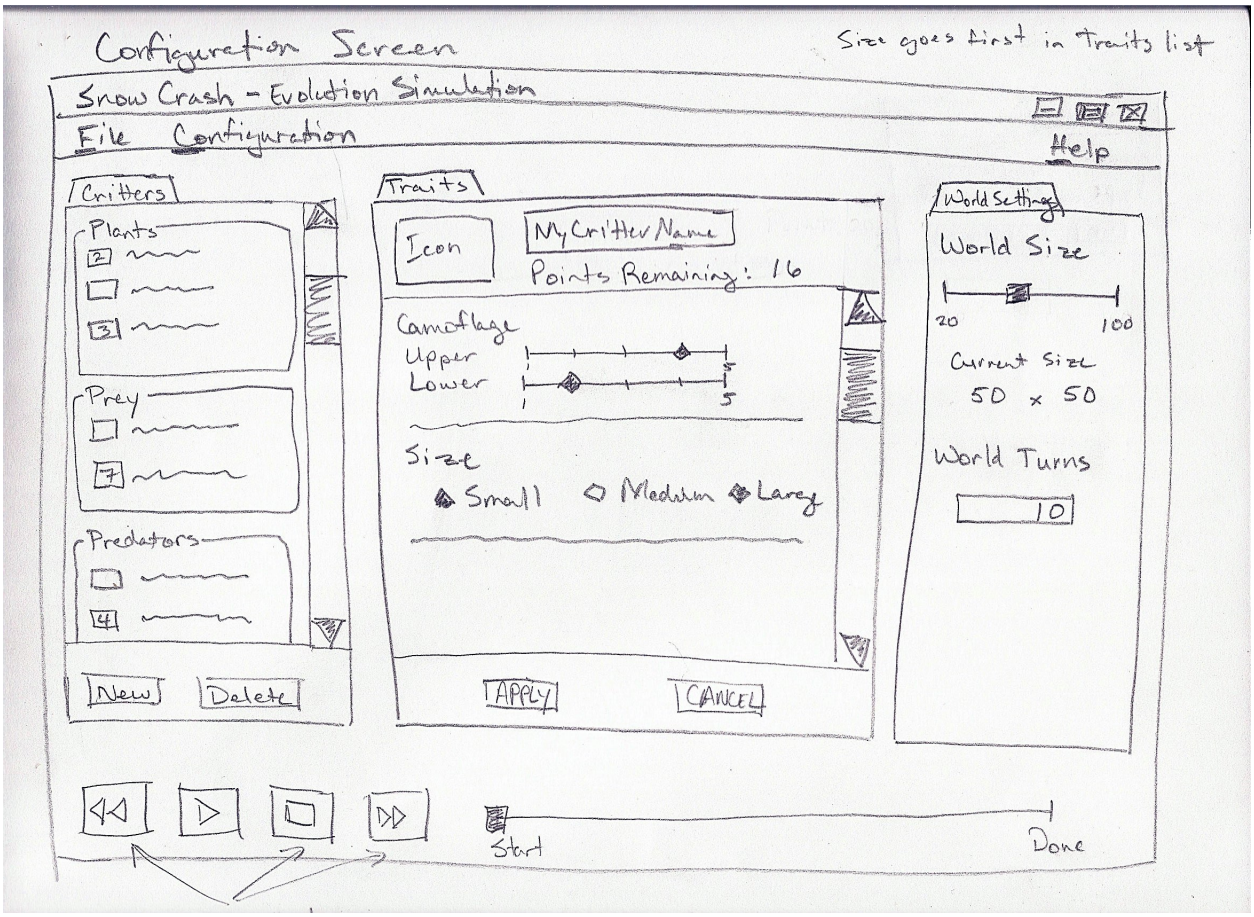
In addition to the generic components listed above in the above section, there are four major panels:

1. The Critters Panel, which list the loaded Critters templates available for a simulation.

2. The Traits Panel, which allow the user to configure the Critter templates.

3. The World Settings Panel, which allow the user to configure the world settings for the simulation.

4. The Play Panel, which starts the simulation.



2.2.3Simulation Screen

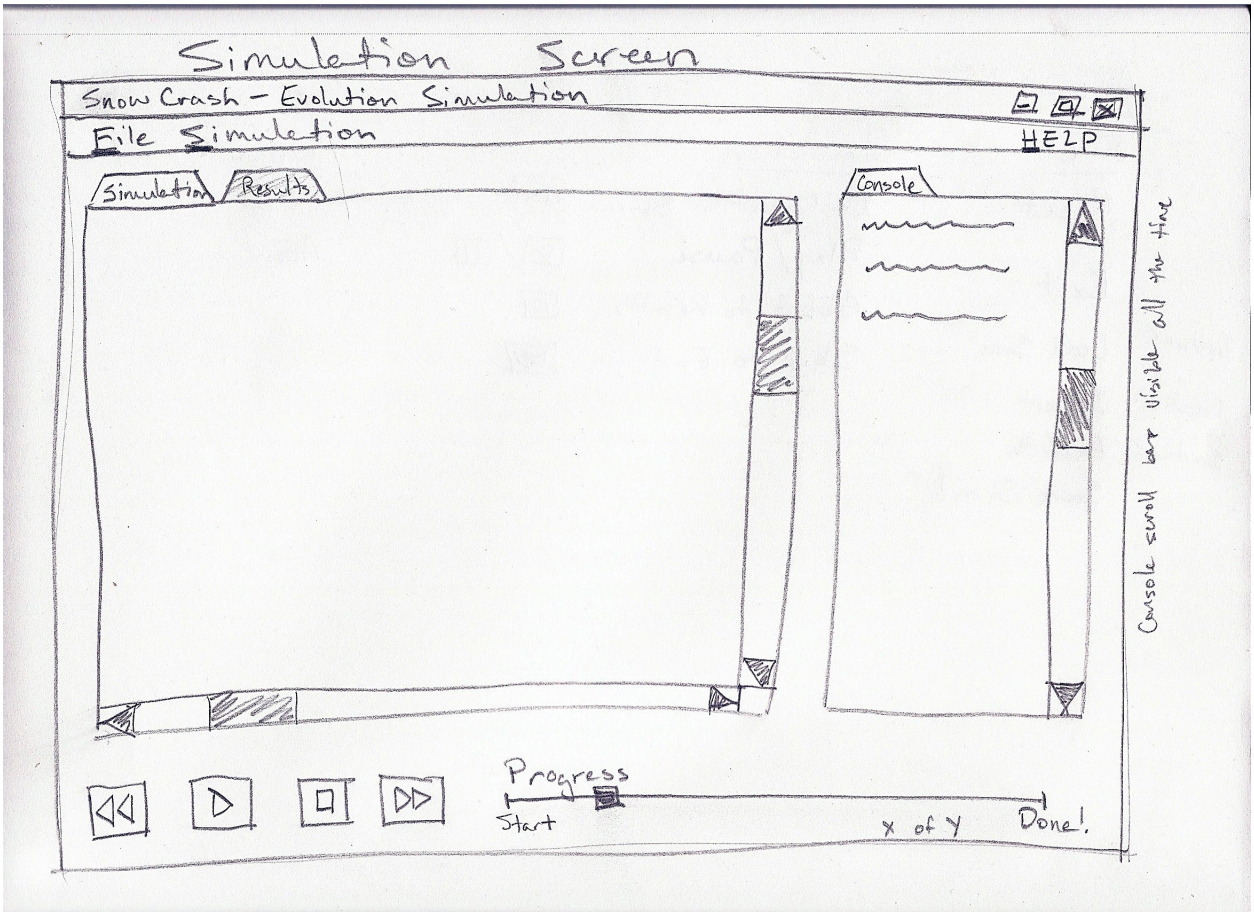
This is the screen which runs the simulation and displays the actions that occur in the simulation world.

In addition to the generic components listed above in the above section, there are three major panels:

1. The Simulation Panel, which shows the running simulation.

2. The Console Panel, which lists major activities that have occurred in the simulation.

3. The Play Panel, which gives the user control over how the simulation runs.



2.2.4Results Screen

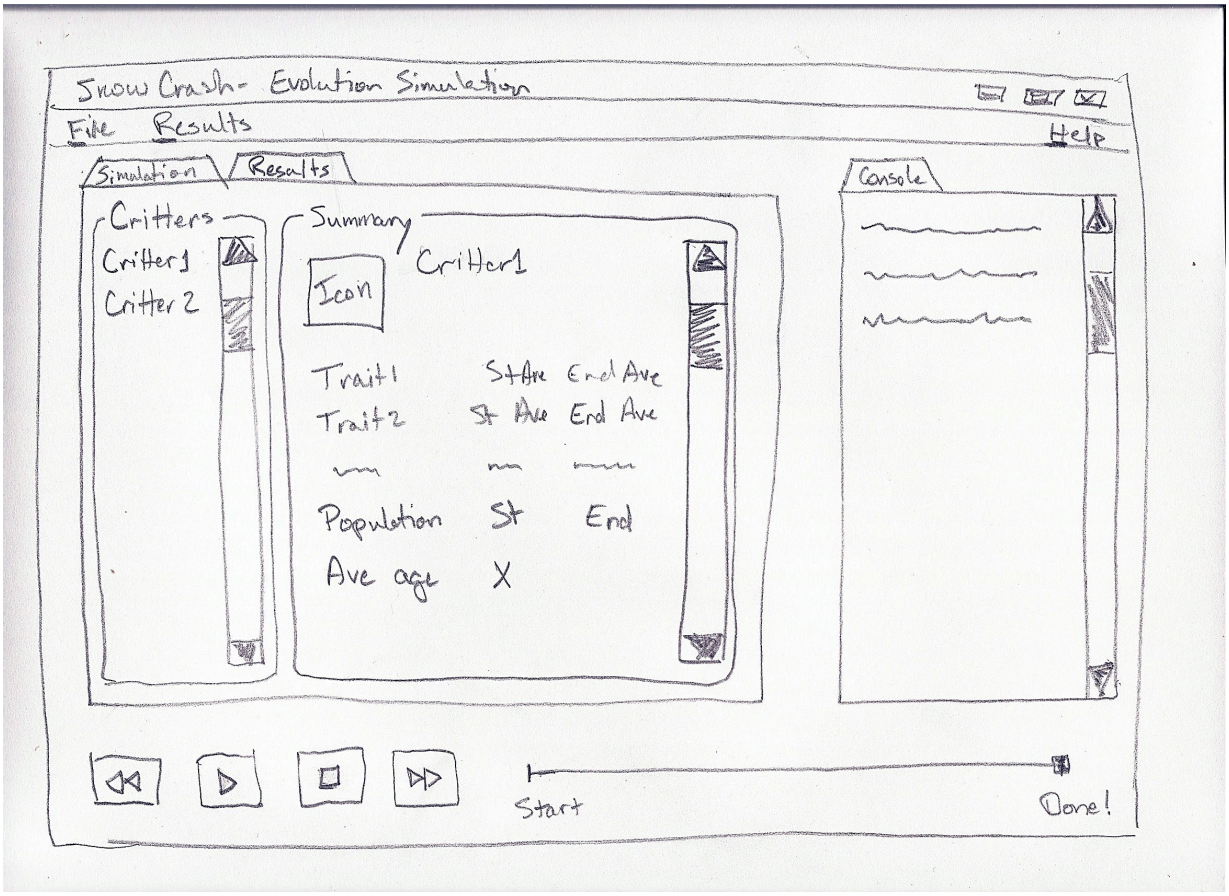
This screen displays the results of the simulation.

In addition to the generic components listed above in the above section, there are three major panels:

1. The Results Panel displays the results, segregated by Critter, and allows the user to view the last turn of the simulation.

2. The Console Panel displays the last 10 turns of the simulation.

3. The Play Panel, which allows the user to return to the Configuration Screen.



2.2.5Menu Items

File
--
Reset

Exit

Configuration
--
New Critter Template
Delete Critter Template

Import Critter Templates
Export Critter Templates

Load Configuration
Load Simulation
Load Results

Save Configuration

Start Simulation

Simulation
--
Play/Pause
Back to Configuration
Abort to Results
Simulate to End

Save Simulation

Results
--
Open Log

Save Results

Help
--
About

3. Functional Model

This section provides usage scenarios for the software. It organizes information collected during requirements elicitation into use-cases. Activity diagrams can also be included to supplement selected use-cases.

3.1 User Profiles

There is only one user profile: user. The user of this software is the person that starts the application, designs critters, runs simulations, and reviews results.

3.2 Use Case Summary

Primary Actor	Use Cases	Name
User	UI001	Import Critter
User	UI002	Export Critter
User	UI003	Modify Traits
User	UI004	Start Simulation
User	UI005	Pause Simulation
User	UI006	Stop/Abort Simulation
User	UI007	Load Simulation
User	UI008	Save Simulation
User	UI009	View Results
User	UI010	Exit Simulation
User	UI011	Reset Simulation

3.3 Use Cases

3.3.1 Import Critter

Use Case ID:	UI001		
Use Case Name:	Import Critter		
Created By:	Jeffrey Dunn	Last Updated By:	Dale
Date Created:	09/26/10	Date Last Updated:	11/01/10

Actors:	User
Description:	The user selects this menu option to import to the current game session a critter that was previously saved to a file.

Trigger:	The user may decide on this use case as an alternative to modifying all critter Traits manually.
Preconditions:	1.Can only be accessed from the menu on the Configuration Screen. 2.A previously exported critter file must exist.
Postconditions:	1.Critter Traits are imported and will update the Configuration Screen accordingly. Pop-up disappears. 2.Return to Configuration Screen
Normal Flow:	1.User selects the menu option to import a critter. 2.A pop-up screen with text field asks the user for the critter filename to import. Pop-up also has an "Okay" button. 3.User types in the path and filename and clicks the "Okay" button. 4.File is read and Traits are loaded into the game. Configuration Screen is updated accordingly.
Alternative Flows:	
Exceptions:	User types in an incorrect or improper filename: Pop-up displays a message "File xxxxx.xxx does not exist. Try again." User tries to import a critter file that has corrupted data: Pop-up displays a message "Error loading xxxxx.xxx."
Includes:	calls UI003 after completion.
Priority:	High
Frequency of Use:	once every 15 minutes
Business Rules:	
Special Requirements:	
Assumptions:	Users may want to run simulations with previous used critter Traits. Users would tire easily of manually modifying all Traits prior to simulation.
Notes and Issues:	

3.3.2Export Critter

Use Case ID:	UI002		
Use Case Name:	Export Critter		
Created By:	Jeffrey Dunn	Last Updated By:	Dale
Date Created:	09/26/10	Date Last Updated:	11/01/10

Actors:	User
Description:	The user selects this menu option to save critter Traits to a file.
Trigger:	The user may decide on this use case if the user wishes to import the existing Traits into a future game session.

Preconditions:	1.Can only be accessed from the menu on the Configuration Screen.
Postconditions:	1.Critter Traits are exported to a file. Pop-up disappears. 2.Return to Configuration Screen
Normal Flow:	1.User selects the menu option to export a critter. 2.A pop-up screen with text field asks the user for the filename of the critter to export. Pop-up also has an "Okay" button. 3.User types in the path and filename and clicks the "Okay" button. 4.File is written with Traits from the current game session. 5.Return to Configuration Screen
Alternative Flows:	
Exceptions:	User types in an improper filename: Pop-up displays a message "Incorrect filename. Try again." User types in a path that does not have write permissions or is out of drive space: Pop-up displays a message "Error saving xxxxx.xxx."
Includes:	calls UI003 after completion.
Priority:	High
Frequency of Use:	once every 15 minutes
Business Rules:	
Special Requirements:	
Assumptions:	Users may want to run simulations with previous used critter Traits. Users would tire easily of manually modifying all Traits prior to simulation.
Notes and Issues:	

3.3.3Modify Traits

Use Case ID:	UI003		
Use Case Name:	Modify Traits		
Created By:	Jeffrey Dunn	Last Updated By:	Dale
Date Created:	09/26/10	Date Last Updated:	11/01/10

Actors:	User
Description:	The user is brought to the Configuration Screen by default when the program starts or when the user manually stops a simulation. Modify Traits is the default case on the Configuration Screen.
Trigger:	The user executes Snow Crash or the user manually stops a simulation from running.

Preconditions:	1.Can only be accessed on the Configuration Screen. 2.User executes Snow Crash OR 3.User manually stops a simulation.
Postconditions:	1.Traits are changed, assuming the user modified them manually.
Normal Flow:	1.User executes Snow Crash. 2.The Configuration Screen appears with attribute settings that the user can interactively modify. 3.User modifies Traits in the GUI 4.Attribute values are updated accordingly.
Alternative Flows:	User first Stops a simulation. The Configuration Screen appears with attribute settings that the user can interactively modify.
Exceptions:	None
Includes:	None
Priority:	High
Frequency of Use:	once every 15 minutes
Business Rules:	
Special Requirements:	
Assumptions:	Users should customize critter Traits before running a simulation.
Notes and Issues:	

3.3.4Run Simulation

Use Case ID:	UI004		
Use Case Name:	Run Simulation		
Created By:	Jeffrey Dunn	Last Updated By:	Dale
Date Created:	09/26/10	Date Last Updated:	11/01/10

Actors:	User
Description:	Runs the simulation for the current attribute settings.
Trigger:	The user presses the “Run Simulation” button on the Customization or Simulation screens.
Preconditions:	1.User is finished modifying Traits on the Configuration Screen OR 2.Simulation is Paused in the Simulation Screen
Postconditions:	1.Traits are changed, assuming the user modified them manually.
Normal Flow:	1.User presses the “Run Simulation” button from the Configuration Screen. 2.The screen switches to the Simulation Screen.

	3.The simulation starts to run with a graphic representation on the screen. A timer is displayed and begins counting elapsed simulation time.
Alternative Flows:	User pauses the simulation from the Simulation Screen. The user presses the "Start/Restart Simulation" button. The simulation resumes from where it was paused.
Exceptions:	None
Includes:	None
Priority:	High
Frequency of Use:	once every 5 minutes
Business Rules:	
Special Requirements:	
Assumptions:	The simulation is the whole point of the game.
Notes and Issues:	

3.3.5Pause Simulation

Use Case ID:	UI005		
Use Case Name:	Pause Simulation		
Created By:	Jeffrey Dunn	Last Updated By:	Dale
Date Created:	09/26/10	Date Last Updated:	11/01/10

Actors:	User
Description:	Pauses a running situation.
Trigger:	The user presses the "Pause" button on the Simulation Screen.
Preconditions:	1.A simulation is running on the Simulation Screen
Postconditions:	1.Simulation pauses. 2.User may unPause the simulation by pressing the "Play" button
Normal Flow:	1.A simulation is running 2.The user presses the "Pause" button. 3.The simulation pauses. 4.The "Play" button can be pressed to resume the simulation at any time.
Alternative Flows:	Instead of unPausing the simulation, the user can stop the simulation by pressing the "Stop" button.
Exceptions:	None
Includes:	None
Priority:	High
Frequency of Use:	once every 5 minutes
Business Rules:	
Special Requirements:	

Assumptions:	The user may have to answer the phone, eat, use the restroom or whatever and would like to pause the simulation.
Notes and Issues:	

3.3.6Stop/About Simulation

Use Case ID:	UI006		
Use Case Name:	Stop/Abort Simulation		
Created By:	Jeffrey Dunn	Last Updated By:	Dale
Date Created:	09/26/10	Date Last Updated:	11/01/10

Actors:	User
Description:	Stop the simulation early.
Trigger:	The user presses the “Stop” button on the Simulation Screen OR the simulation runs its course and completes on its own.
Preconditions:	1.A simulation is running or Paused on the Simulation Screen
Postconditions:	1.Depends on the condition by which the simulation stops. If the “Stop” button was pressed by the user, then the screen switches to the Configuration Screen. If the simulation stopped on its own, the screen switches to the Results Screen.
Normal Flow:	1.A simulation is running 2.The simulation finishes on its own and a “Simulation Complete” message is displayed. 3.“Stop” and “Pause” buttons are inactive. 4.The screen switches to the Results Screen.
Alternative Flows:	The user presses the “Stop” button while a simulation is running or paused on the Simulation Screen. The screen then switches to the Configuration Screen.
Exceptions:	None
Includes:	UI004 or UI005 is a precondition. UI003 or UI009 result, depending on what caused the simulation to stop.
Priority:	High
Frequency of Use:	once every 15 minutes
Business Rules:	
Special Requirements:	
Assumptions:	The user may want to restart the simulation or simply view results up to this point. The user may decide to stop the game and turn off the computer because of real life priorities. Maybe he has to walk the dog, go to class or pay attention to his girlfriend.
Notes and Issues:	

3.3.7Load Simulation

Use Case ID:	UI007		
Use Case Name:	Load Simulation		
Created By:	Jeffrey Dunn	Last Updated By:	Dale
Date Created:	09/26/10	Date Last Updated:	11/01/10

Actors:	User
Description:	Loads simulation results that were saved from a previous session.
Trigger:	The user selects the “Load Simulation” menu option from the Configuration Screen.
Preconditions:	1.The current screen is the Configuration Screen (UI003)
Postconditions:	1.The results are displayed in the Results Screen
Normal Flow:	1.The user is in the Configuration Screen. 2.The User selects the “Load Simulation” menu option. 3.A Pop-up window appears with a text field and asks the user to input a filename. there is also an “Okay” button. 4.The user types in the filename and clicks the “Okay” button. 5.The file is loaded into the current session. 6.The Pop-up disappears 7.The main window changes to the Simulation Screen and displays the results loaded from the results-file.
Alternative Flows:	None
Exceptions:	User types in an incorrect or improper filename: Pop-up displays a message “File xxxxx.xxx does not exist. Try again.” User tries to load a file that has corrupted data: Pop-up displays a message “Error loading xxxxx.xxx.”
Includes:	UI003 is a precondition. UI009 is included in the steps
Priority:	High
Frequency of Use:	once every 15 minutes
Business Rules:	
Special Requirements:	
Assumptions:	The user may wish to view simulation results from a previous game session.
Notes and Issues:	

3.3.8Save Simulation

Use Case ID:	UI008
Use Case Name:	Save Simulation

Created By:	Jeffrey Dunn	Last Updated By:	Dale
Date Created:	09/26/10	Date Last Updated:	11/01/10

Actors:	User
Description:	Saves simulation from the current session.
Trigger:	The user selects the "Save Simulation" menu option from the Results Screen.
Preconditions:	1.The current screen is the Results Screen (UI009)
Postconditions:	1.The simulation results from the current session are saved to a file.
Normal Flow:	1.The current screen is the Results Screen 2.The user selects the "Save Simulation" menu option. 3.A pop-up screen with text field asks the user for the filename of the simulation results to be saved. Pop-up also has an "Okay" button. 4.User types in the path and filename and clicks the "Okay" button. 5.File is written with simulation results from the current game session. 6.The Pop-up disappears 7.Returned to the Results Screen.
Alternative Flows:	None
Exceptions:	User types in an improper filename: Pop-up displays a message "Incorrect filename. Try again." User types in a path that does not have write permissions or is out of drive space: Pop-up displays a message "Error saving xxxxx.xxx."
Includes:	UI009 is included in the steps
Priority:	High
Frequency of Use:	once every 15 minutes
Business Rules:	
Special Requirements:	
Assumptions:	The user may wish to save the simulation for later viewing.
Notes and Issues:	

3.3.9View Results

Use Case ID:	UI009		
Use Case Name:	View Results		
Created By:	Jeffrey Dunn	Last Updated By:	Dale
Date Created:	09/26/10	Date Last Updated:	11/01/10

Actors:	User
Description:	Shows the Simulation results
Trigger:	The user selects the “Save Simulation” menu option from the Results Screen.
Preconditions:	1.The user stops a simulation early from the Simulation Screen OR the user chooses the Load Simulation results menu option or View Simulation Results menu option from the Configuration Screen
Postconditions:	1.The user may return to the Configuration Screen after viewing simulation results.
Normal Flow:	1.The user stops a simulation early from the Simulation Screen OR the user chooses the Load Simulation results menu option or View Simulation Results menu option from the Configuration Screen
Alternative Flows:	None
Exceptions:	None
Includes:	User has the option to return to UI003 as a postcondition.
Priority:	High
Frequency of Use:	once every 15 minutes
Business Rules:	
Special Requirements:	
Assumptions:	The user will want to view the simulation results and statistics.
Notes and Issues:	

3.3.10Exit Simulation

Use Case ID:	UI010		
Use Case Name:	Exit Simulation		
Created By:	Jeffrey Dunn	Last Updated By:	Dale
Date Created:	09/26/10	Date Last Updated:	11/01/10

Actors:	User
Description:	Exits the Simulation.

Trigger:	The user selects the “Quit” menu option from the Results Screen.
Preconditions:	1.The Snow Crash program is running
Postconditions:	1.The Snow Crash program is no longer running
Normal Flow:	1.User selects the “quit” menu option at any point from any screen in the program.
Alternative Flows:	None
Exceptions:	None
Includes:	None
Priority:	High
Frequency of Use:	once every 15 minutes
Business Rules:	
Special Requirements:	
Assumptions:	The user may decide to stop the game and turn off the computer because of real life priorities. Maybe he has to walk the dog, go to class or pay attention to his girlfriend.
Notes and Issues:	

3.3.11Reset Simulation

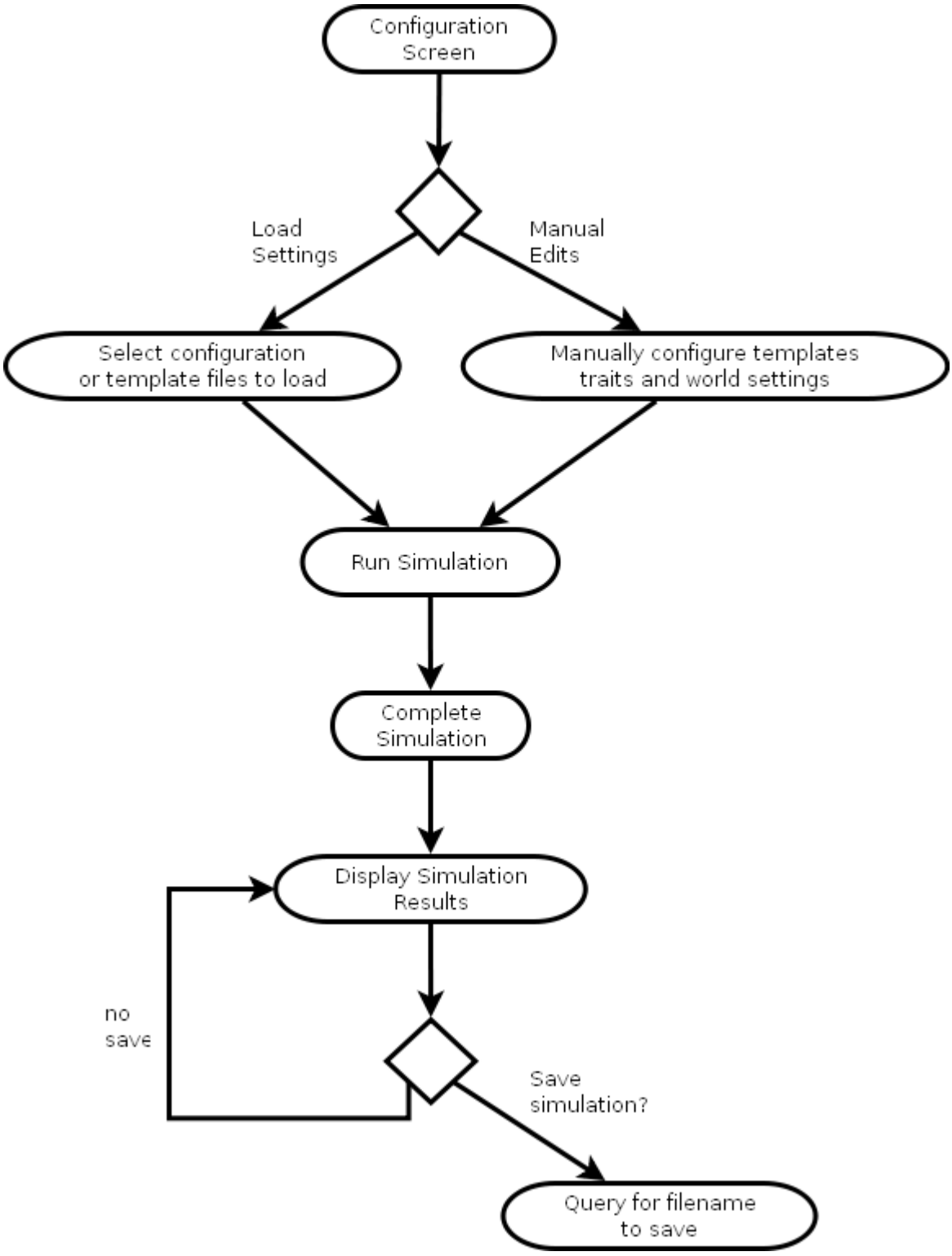
Use Case ID:	UI011		
Use Case Name:	Reset Simulation		
Created By:	Jeffrey Dunn	Last Updated By:	Dale
Date Created:	09/26/10	Date Last Updated:	11/01/10

Actors:	User
Description:	Resets the simulation.
Trigger:	The user selects the “Reset” simulation button from the Simulation Screen.
Preconditions:	1.The simulation is paused or running in the Simulation Screen.
Postconditions:	1.The simulation results are cleared. The screen switches to the Configuration Screen.
Normal Flow:	1.The simulation is paused or running in the Simulation Screen. 2.The user presses the “Reset” button. 3.The simulation ends and the screen switches to the Configuration Screen.
Alternative Flows:	None
Exceptions:	None
Includes:	None
Priority:	High
Frequency of Use:	once every 15 minutes

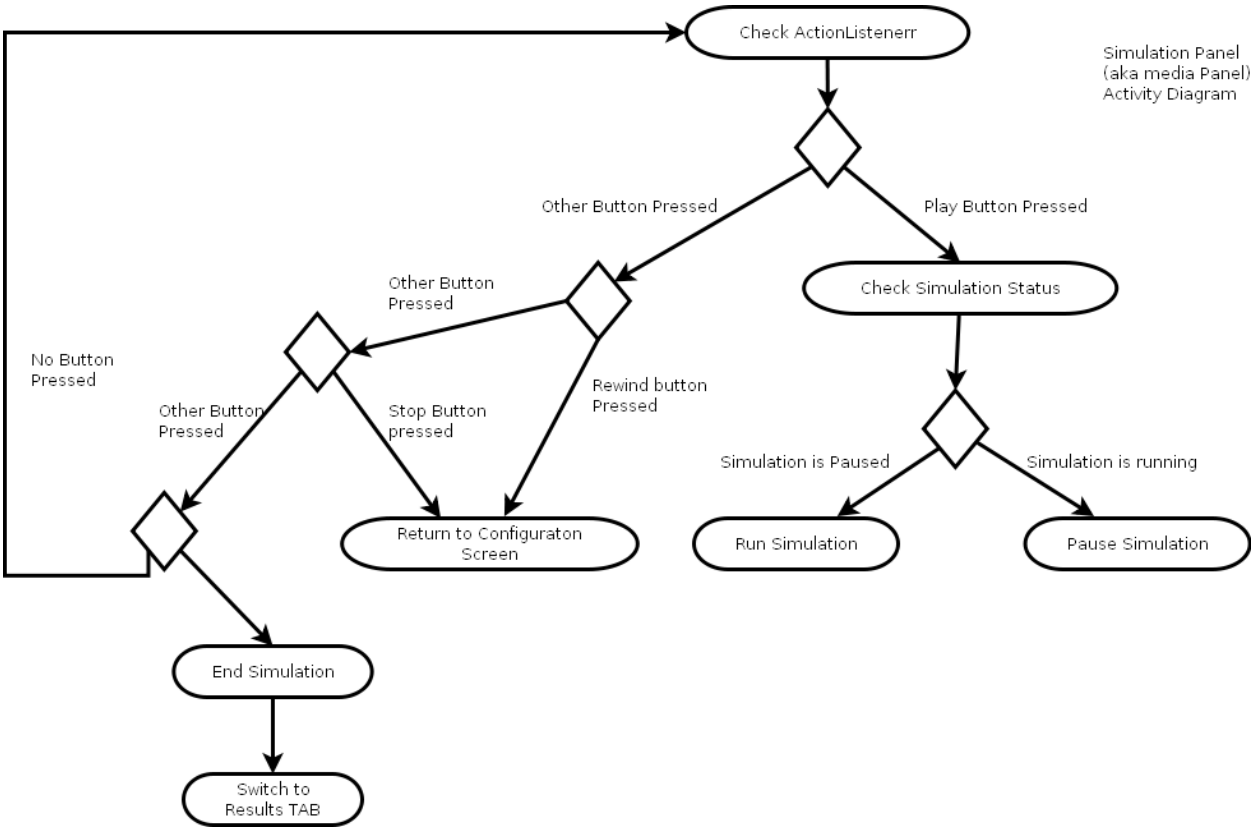
Business Rules:	
Special Requirements:	
Assumptions:	The user may wish to restart the simulation.
Notes and Issues:	

3.4 Activity Diagrams

3.4.1Configuration Activity Diagram



3.4.2Simulation Activity Diagram



4. Structural Model

This section describes the structural model of our project.

4.1 Class/Object List

4.1.1GUI

- GraphicalUserInterface
- MenuBar
- SimulationPanel
- ConfigurationScreen
- ConsolePanel
- SimulationScreen
- ResultsScreen
- CrittersPanel
- TraitsPanel
- WorldSettings

4.1.2Command and Control Engine

- Command
- CommandMediator
- FileManager
- WorldEngine
- ConfigurationManager
- TimingEngine
- DataSource

4.1.3File Manager

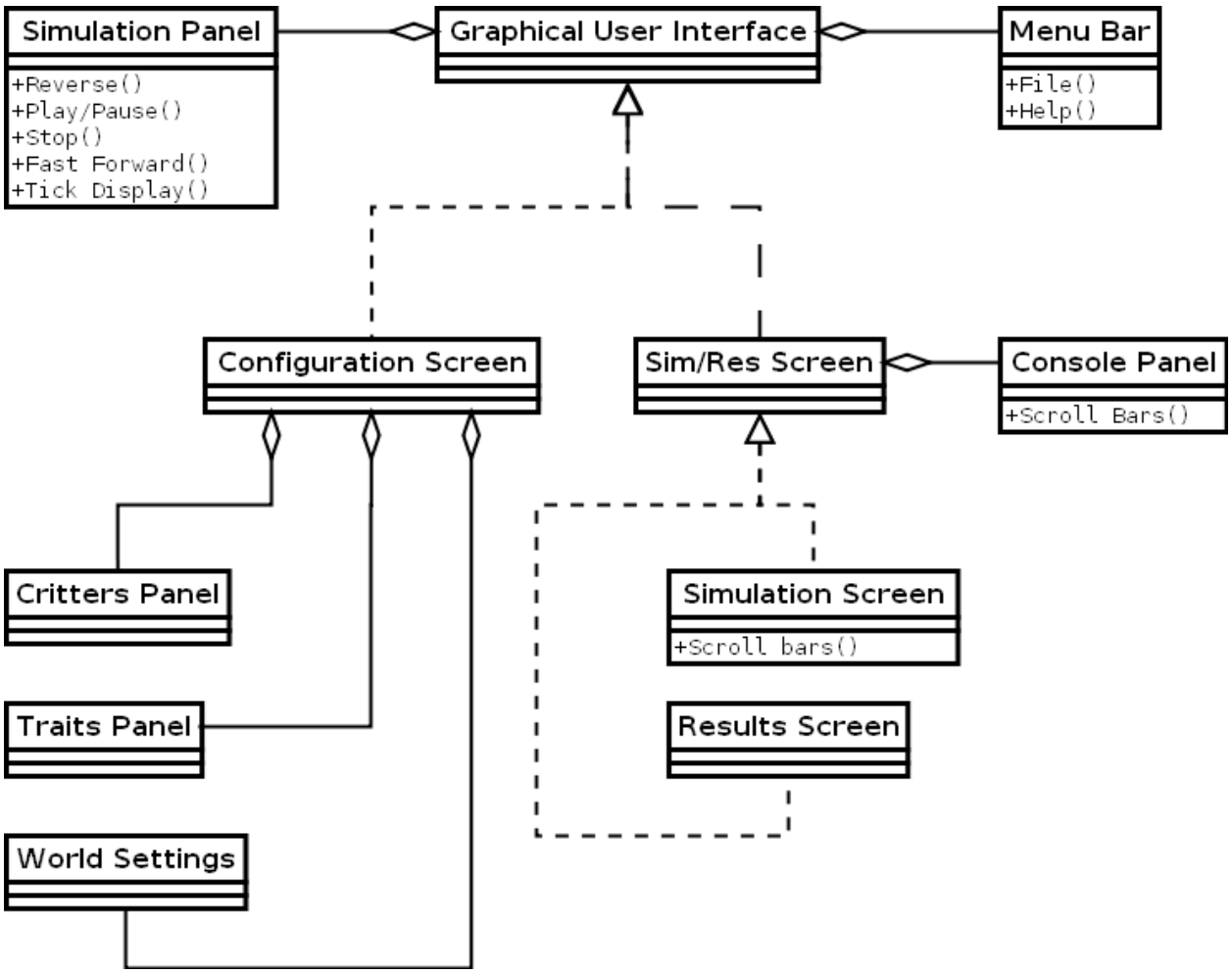
- FileManager
- LogFileViewer

4.1.4Critter Model

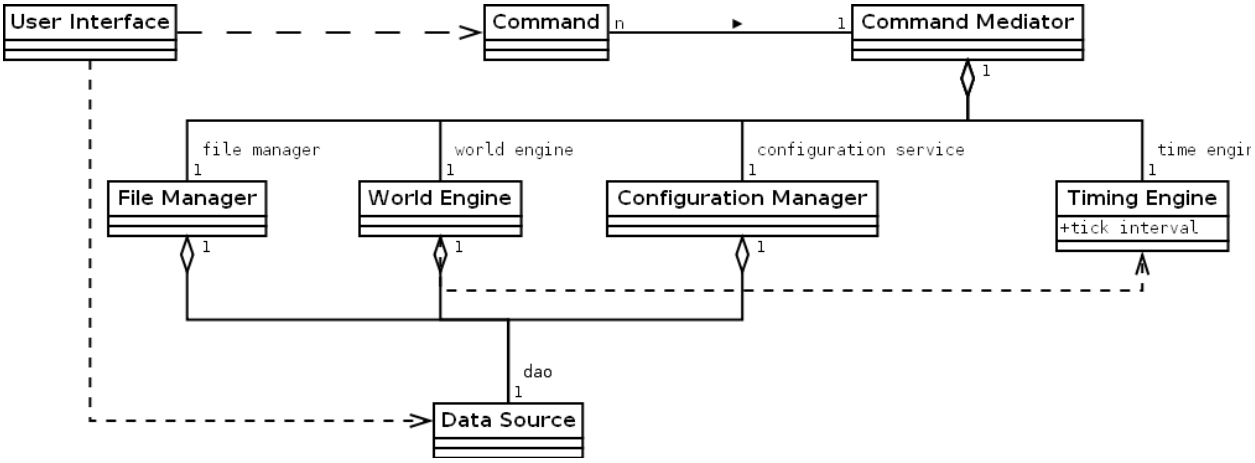
- World
- Critter
- CritterPrototype
- CritterTemplate
- Trait
- Pair
- StateContext
- State
- StateBase
- Hunting
- Moving
- Growing
- Reproducing

4.2 Class Diagrams

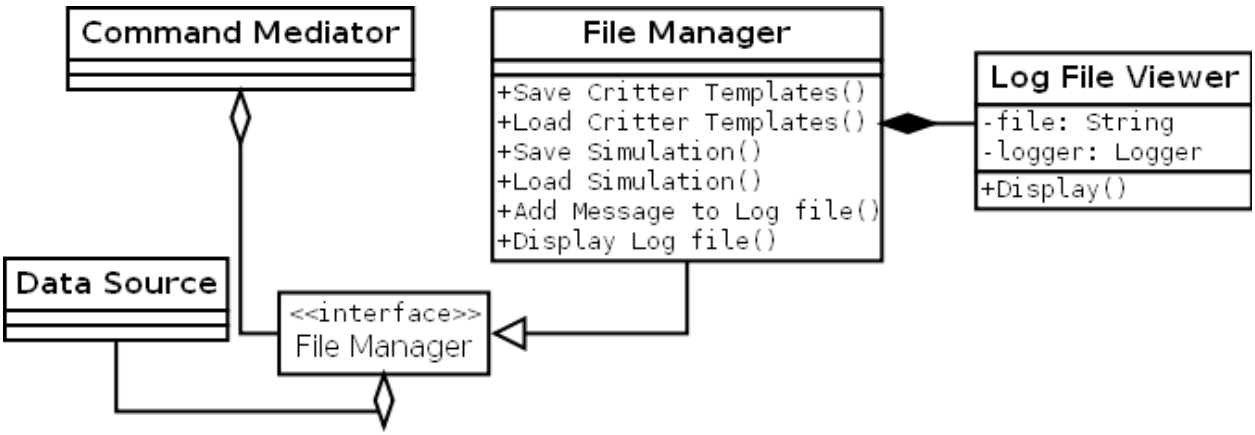
4.2.1GUI Diagram



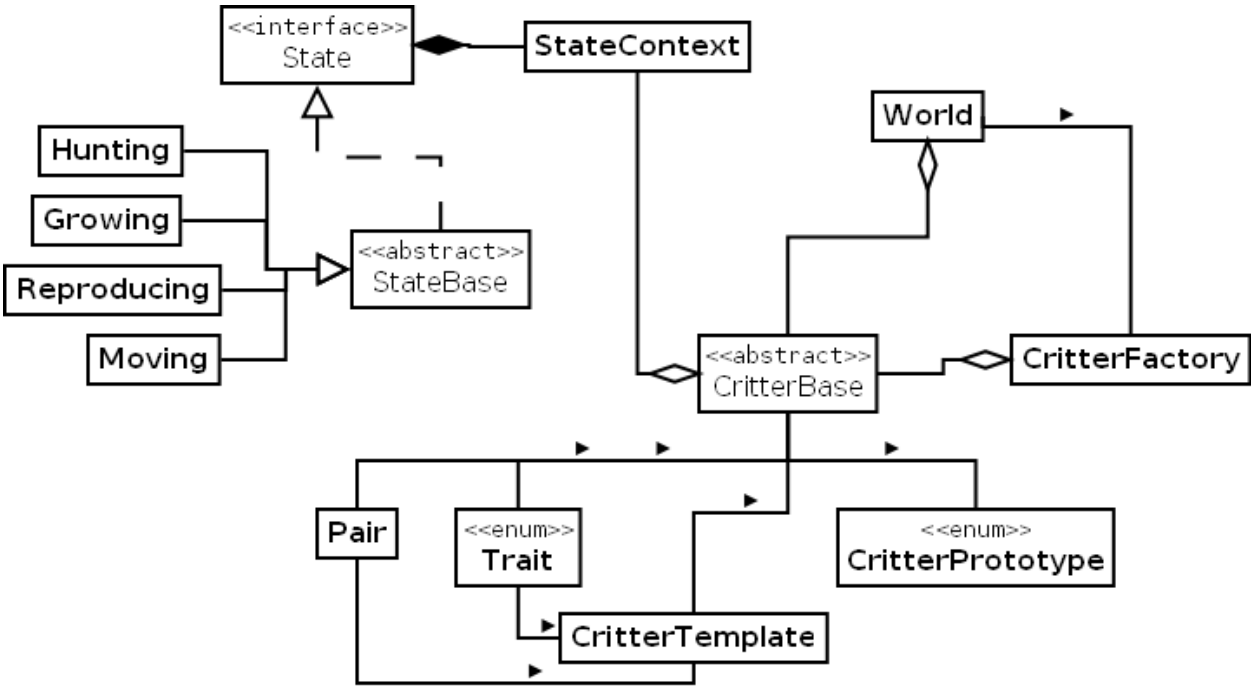
4.2.2Command and Control Engine Diagram



4.2.3File Manager Diagram



4.2.4World, Critter, and Critter Behavioral Diagram



5. Behavioral Model and Description

A description of the internal behavior (i.e. dynamic view) of the software system is presented.

5.1Description for software behavior

5.1.1States

Screen States – The Screen State Diagram describes screen flow.

Plant Behavior – The plant behavior diagram describes how plants behave and their states. Plants grow and reproduce. The mechanics behind this are described in the functional requirements.

Prey and Predator Behavior – The prey and predator behavior diagram describes how prey and predators behave and their states. Prey eat plants and predators eat prey. Specifics behind this are described in the functional requirements.

5.1.2Sequences

Configuration Sequence Diagram – This diagram describes how user interactivity initiates causes changes on the configuration screen.

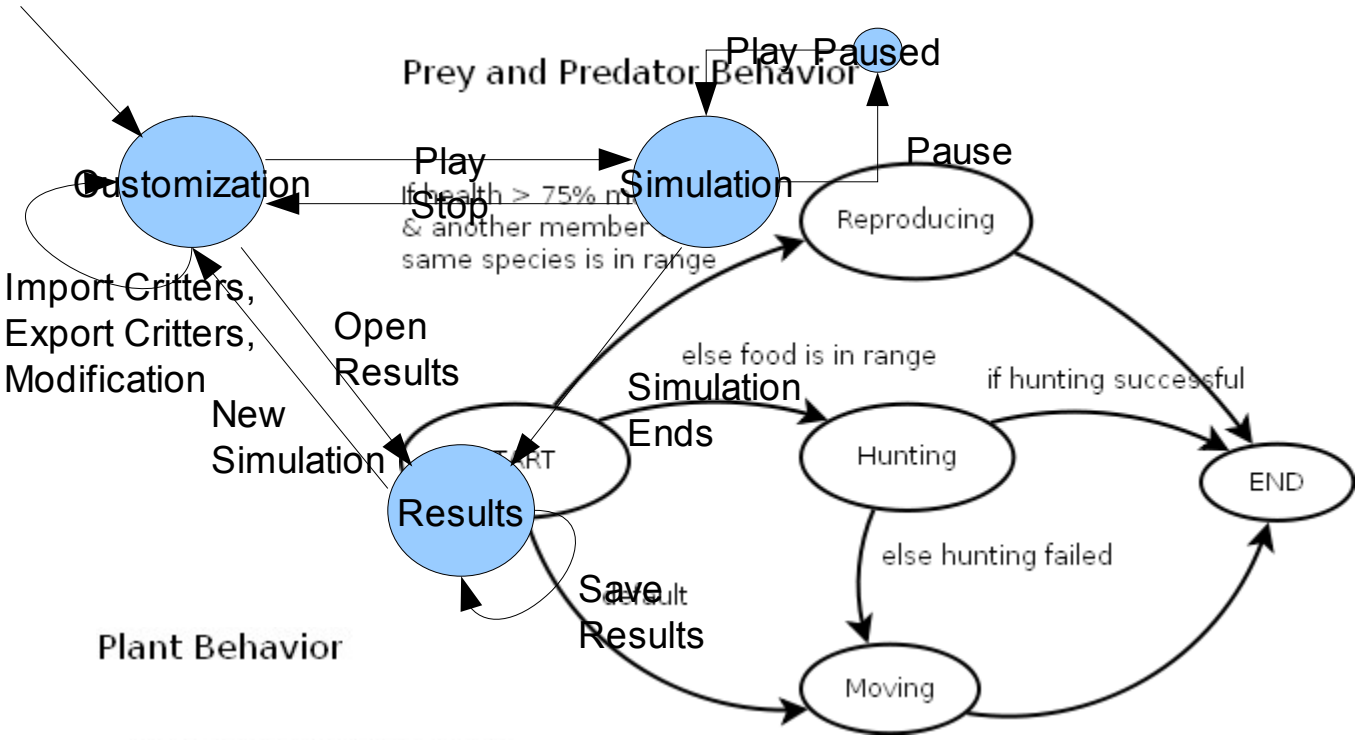
File I/O Sequence Diagram – This diagram describes how the user initiates writing files to disk.

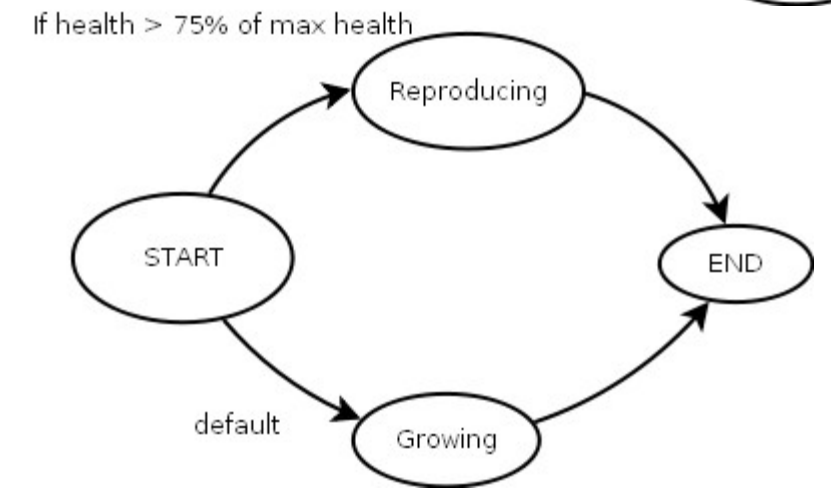
Simulation Timing Diagram – This diagram describes how the user interactivity causes changes on the simulation screen.

File Manager Diagram– The diagram describes how user interactivity opens the results and logs.

5.2 State Transition Diagrams

Screen State Diagram

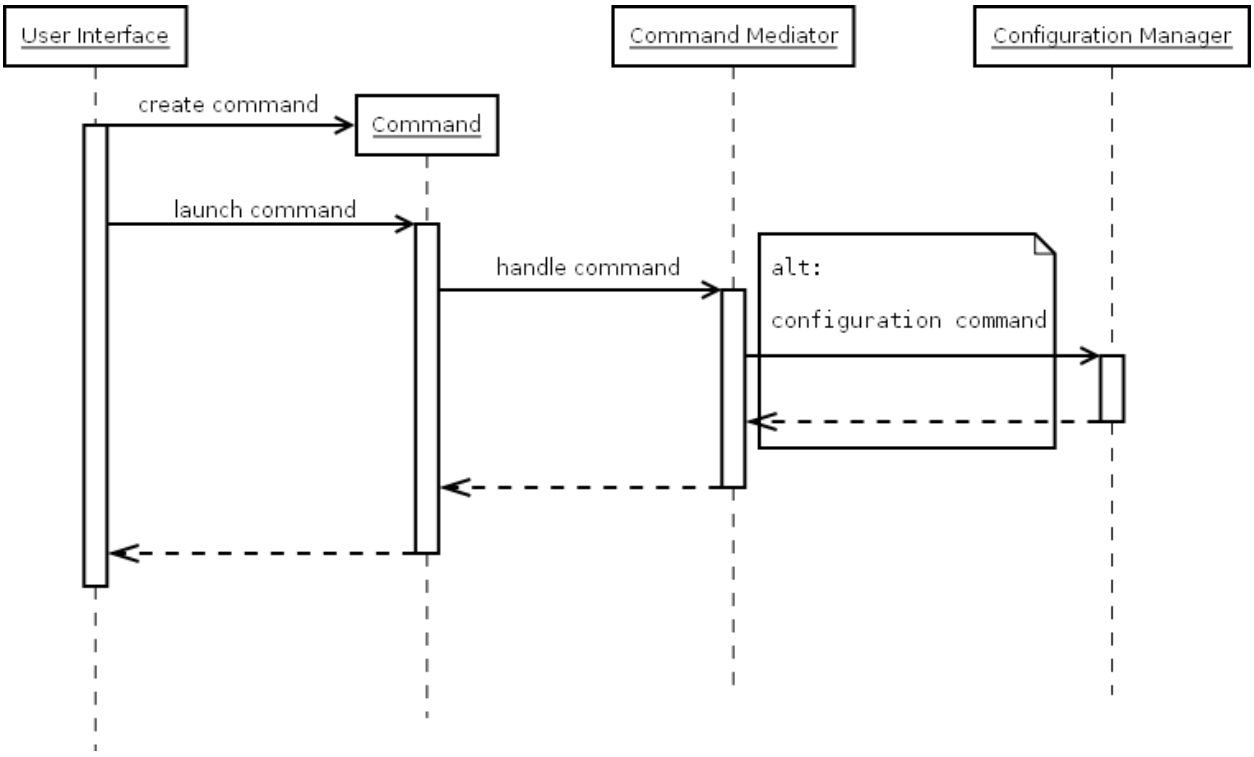




5.3Sequence Diagrams

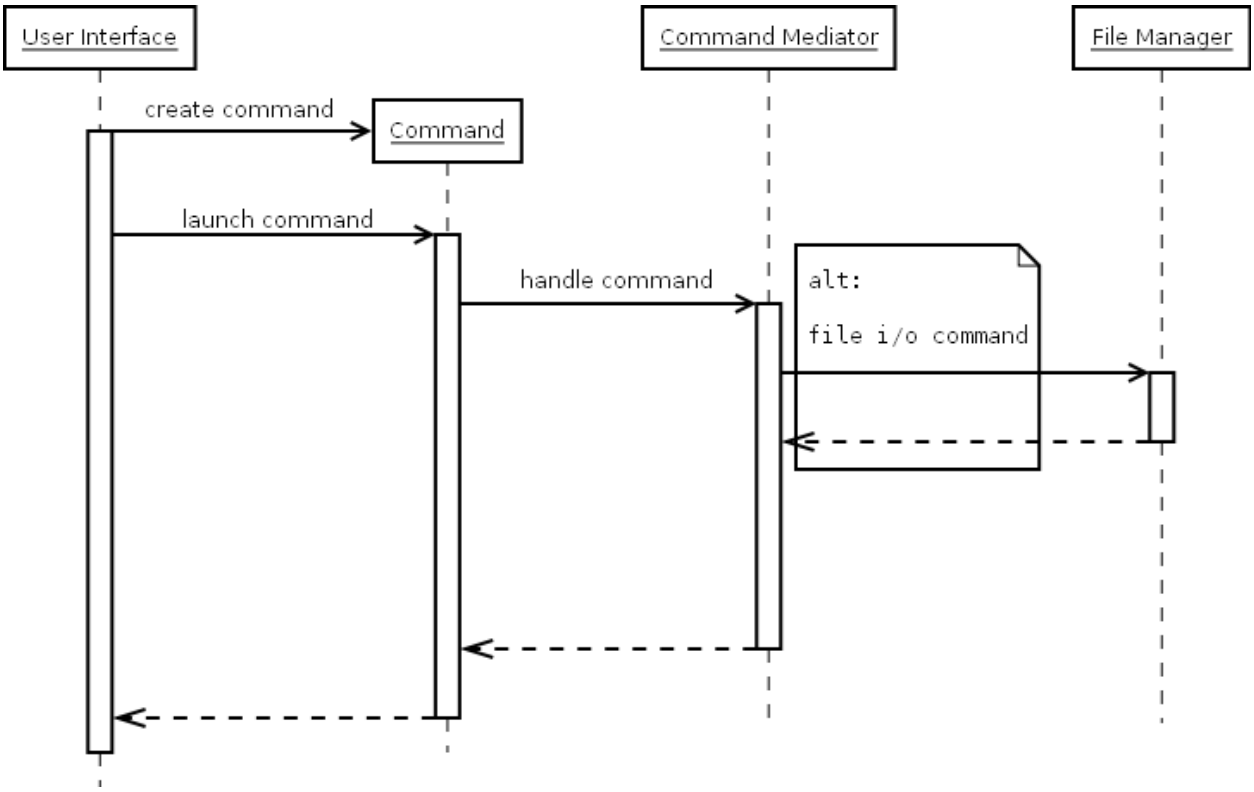
5.3.1Configuration Sequence Diagrams

Covers Use Cases: UI001, UI003, UI007, UI010.



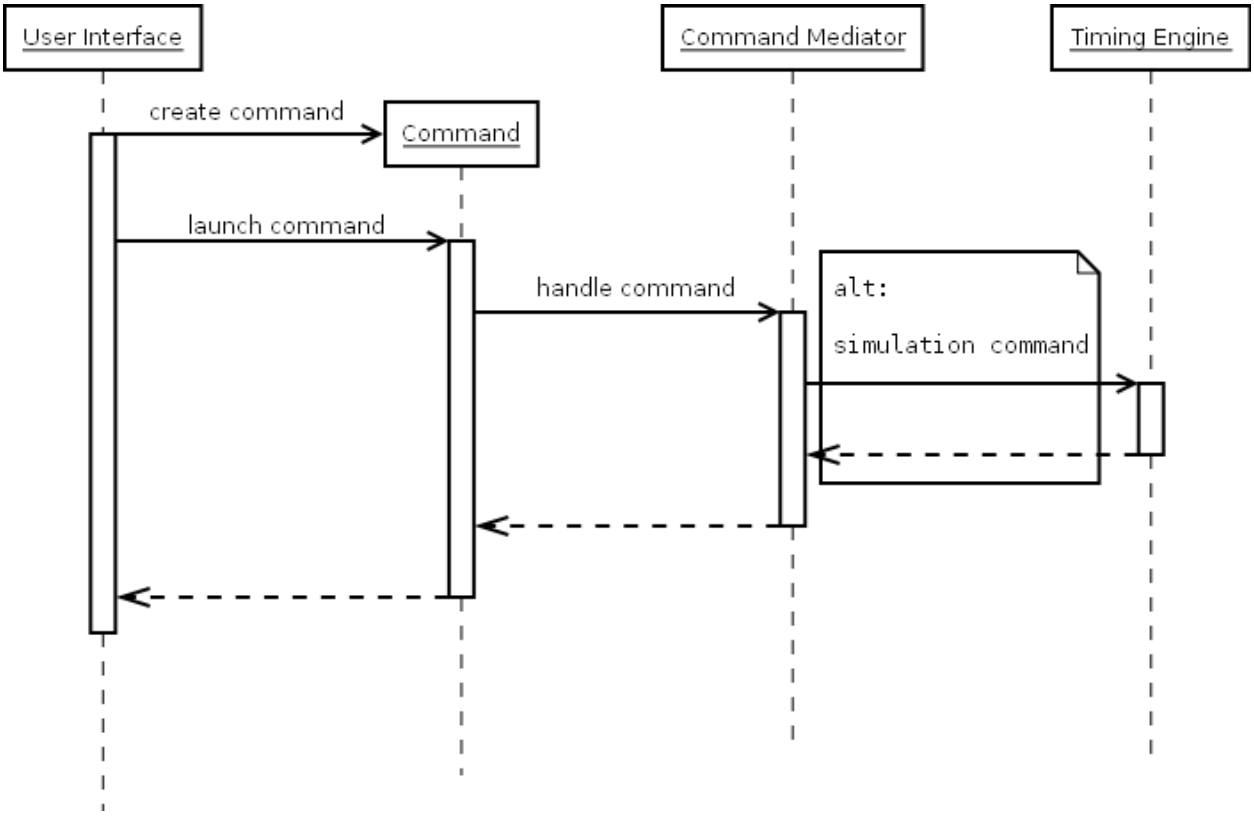
5.3.2File I/O Sequence Diagrams

Covers Use Cases: UI002, UI010.



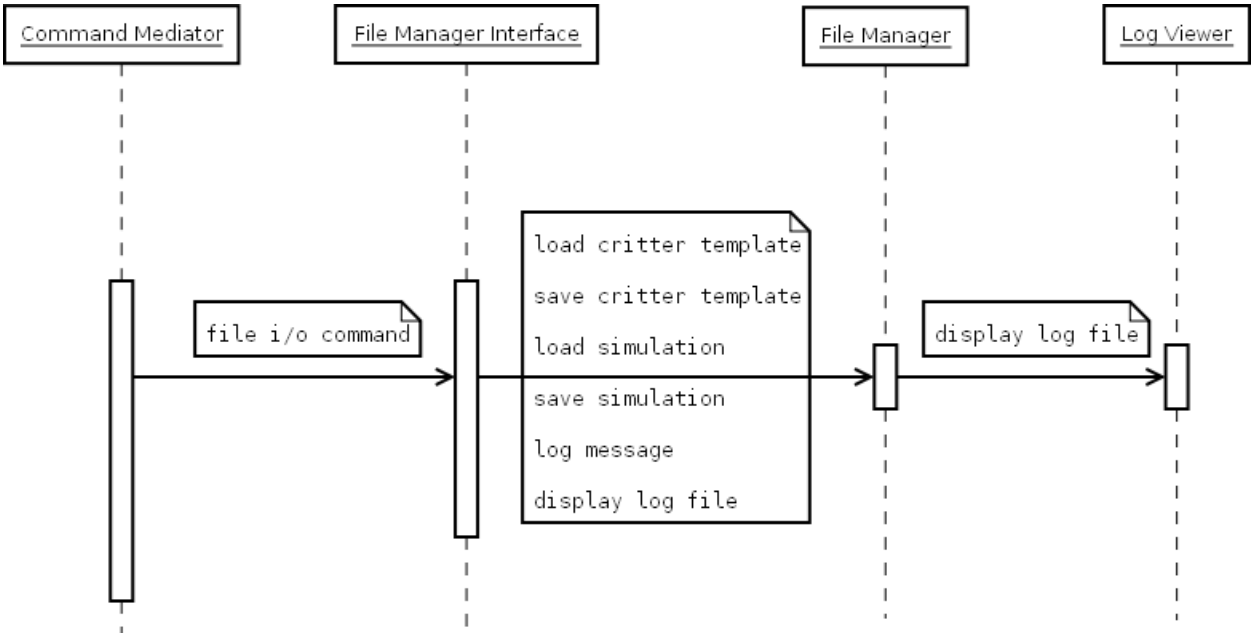
5.3.3Simulation Timing Diagram

Covers Use Cases: UI004, UI005, UI006, UI008, UI010, UI011.



5.3.4File Manager Sequence Diagram

Covers Use Cases: UI009.



6. Requirements Definition

Each requirement is defined by:

- Req. #: This is a unique id for this requirement. Functional requirements are assigned #'s based on their Point Release. Non-functional requirements are numbered separately starting from 1.
- PR: Point Release level
  - 0.0: Base-lined requirements
  - 0.1: Configuration screen requirements
  - 0.2: Simulation screen requirements
  - 0.3: Results screen requirements
- Category: Type of requirement

- File I/O: File Input/Output requirement
- General: Overall requirement
- Mechanics: Simulation mechanics requirement
- UI: User Interface requirement
- USE CASE: Use case defined requirement
- NON-FUNCT: Non-functional requirement

• Requirement: The text of the requirement.

6.1Functional Requirements

Req. #	PR	Category	Requirement
101	0.1	USE CASE	The configuration screen shall allow the user to create critter templates.
102	0.1	USE CASE	The configuration screen shall allow the user to modify critter template traits.
103	0.1	USE CASE	The configuration screen shall allow the user to export all current critter templates.
104	0.1	USE CASE	The configuration screen shall allow the user to import saved critter templates from an earlier session.
106	0.1	USE CASE	The configuration screen shall allow the user to load the saved results of a completed simulation.
107	0.1	USE CASE	The configuration screen shall allow the user to name critter templates.
108	0.1	USE CASE	The software shall provide a help menu for the user to reference.
109	0.1	USE CASE	The configuration screen shall allow the user to reset to default configurations.
110	0.1	USE CASE	The software shall provide a means for exiting the application.
111	0.1	USE CASE	The configuration screen shall allow the user to configure the size of the simulation world and the duration of the simulation (in ticks).
112	0.1	USE CASE	The configuration screen shall allow the user to include a subset of all critter templates to be used in the simulation.
113	0.1	USE CASE	The configuration screen shall allow the user to apply or cancel changes to critter template traits.
114	0.1	USE CASE	The configuration screen shall allow the user to delete critter templates.
115	0.1	FILE I/O	The import and export functionality shall use the JSON format.

116	0.1	FILE I/O	The import functionality shall rename duplicated critter templates by appending a number (1, then 2, then 3, etc.).
117	0.1	UI	The configuration screen shall consist of four parts: the critter template list, the critter template traits panel, the world configuration panel, and the simulation panel.
118	0.1	UI	The simulation panel shall consist of a Play/Pause button, a Stop button, and a timer that shows the current tick and full duration (in ticks).
119	0.1	UI	The configuration screen shall enforce trait ranges by using sliders and/or combo boxes.
120	0.1	UI	All panels shall provide scroll bars, as necessary, when there is more information to display than real estate to display it in.
121	0.1	UI	The critter template traits panel shall display each of the critter template's traits and the icon to be used on the simulation map.
122	0.1	UI	The critter template traits panel and the world configuration panel shall have an Apply button and a Cancel button.
123	0.1	UI	The window-size of the program shall be adjustable from a minimum size of 800x600 pixels
124	0.1	MECHANICS	The software shall recognize the following critter template traits: size (large, medium, or small), vision (1-5), speed (1-5), camouflage (1-5), combat (1-5), endurance (1-5), and age.
125	0.1	MECHANICS	Each critter template shall have a minimum and maximum bound for each trait.
126	0.1	MECHANICS	The software shall define an allocation pool for assigning trait values to a critter template.
127	0.1	MECHANICS	Each critter's allocation pool starts at 20 points.
128	0.1	MECHANICS	Each trait starts at one; moving the slider to the left one unit costs one allocation point, up to four points.
129	0.1	MECHANICS	The software shall support three critter prototypes: plants, prey, and predators.
120	0.1	GENERAL	The software shall support the English language.
201	0.2	USE CASE	The configuration screen shall allow the user to start a simulation.
202	0.2	USE CASE	The simulation screen shall allow the user to pause a simulation.
203	0.2	USE CASE	The simulation screen shall allow the user to resume a paused simulation.
204	0.2	USE CASE	The simulation screen shall allow the user to go back to the configuration screen. This aborts the current simulations.
205	0.2	USE CASE	The simulation screen shall allow the user to stop a simulation. This immediately takes the user to the results screen without completing the simulation.

206	0.2	USE CASE	The simulation screen shall allow the user to skip to the end of the simulation. This simulates the remainder of the simulation per the user's input on the configuration screen. It then proceeds to the results screen.
207	0.2	USE CASE	The simulation screen shall allow the user to save the active simulation.
208	0.2	USE CASE	When the user saves a simulation, the software shall pause the application before saving.
209	0.2	USE CASE	When the simulation completes, the software shall advance to the results screen.
210	0.2	USE CASE	When an existing simulation is loaded from relevant screens, the software shall present the user with the simulation in the simulation screen; and the simulation shall be in the paused state.
211	0.2	USE CASE	The configuration screen shall allow the user to load a saved simulation.
212	0.2	FILE I/O	The save simulation functionality shall capture all world and critter data, and all active critter templates.
213	0.2	FILE I/O	The load simulation functionality shall restore all world and critter data, and all previously active critter templates.
214	0.2	UI	The simulation screen shall consist of three parts: the world view, the console view, and the simulation panel.
215	0.2	UI	The simulation panel shall appear and function the same on each screen.
216	0.2	UI	The console view shall timestamp each message with the tick number.
217	0.2	UI	The console view shall log the following events: eating, reproduction, and starvation.
218	0.2	UI	The software shall maintain only events from the last ten ticks.
219	0.2	UI	The world view shall display a grid with sprites representing each of the critter templates defined by the user.
220	0.2	UI	All buttons that display an image shall have tooltips.
221	0.2	MECHANICS	Plants shall regenerate health every tick.
222	0.2	MECHANICS	Critters shall perform one of the following actions every tick: eat, reproduce, move.
223	0.2	MECHANICS	When a predator eats a prey, the prey shall be removed and the predator shall take the prey's location. If the predator does not eat the prey, they shall retain their pre-tick locations. It gains the health value of the Prey, up to its own max health.
224	0.2	MECHANICS	When two critters reproduce, the offspring shall be placed at an adjacent location, if one is available. If no adjacent locations are available, the critters cannot reproduce.
225	0.2	MECHANICS	The software shall restrict the world to a square.

226	0.2	MECHANICS	The software shall restrict the population of the world to 70% of the total spaces in the world when initialized.
227	0.2	MECHANICS	The default size of the world is 50x50. The max size of the world is 100x100. The minimum size of the world is 20x20.
228	0.2	MECHANICS	The minimum number of critter templates per world is 1. The minimum number of critters per world is 2 per critter template except for plants which can be 1.
229	0.2	MECHANICS	Each critter will fit in one square. Only one critter may occupy a square.
230	0.2	MECHANICS	Initial critter populations are placed randomly throughout the world on initialization.
231	0.2	MECHANICS	Plant state logic is as follows: If a plant has > 75% health and the reproducing cooldown is up, then it reproduces. If there is another plant in range, the plant will reproduce with that other plant, else it reproduces on their own. Else it grows.
232	0.2	MECHANICS	Prey state logic is as follows: If a prey has > 75% health and another prey is within range with > 75% health, then they reproduce. Else if there is a Plant within range and it's health is below 75%, the Prey eats the Plant. If there is no Plant in range, it moves in a random direction as far as it can.
234	0.2	MECHANICS	Predator state logic is as follows: If a Predator has > 75% health and another Predator is within range with > 75% health, then they reproduce. Else if there is a Prey within range and it's health is below 75%, the Predator eats the Prey. If there is no Prey in range, it moves in a random direction as far as it can.
235	0.2	MECHANICS	When a Prey eats a Plant, it does not have to eat the entire Plant and can simply deduct as much health as it eats. If the Plant dies due to being eaten, then the Prey takes the place of the Plant on the world. If the Plant does not die and there is a free space adjacent to it, then the Prey takes the closest space and occupies it. If there is no adjacent space, the Prey does not eat and remains where it is.
236	0.2	MECHANICS	When a Plant self-reproduces, it creates a clone of itself.
237	0.2	MECHANICS	When a critter reproduces with another of it's species, each critter passes on one random value of every trait pair to the offspring.
238	0.2	MECHANICS	When critters reproduces, the parents sacrifice half their health to produce an offspring and the offspring starts with 75% of the sacrificed health.
239	0.2	MECHANICS	The mechanics of whether a Predator catches a Prey are a function of spotting the Prey and then catching and defeating it, modified by the critters' constitution, vision, speed, combat, and size.
240	0.2	MECHANICS	Critters can only reproduce with critters of the same template.
241	0.2	MECHANICS	The default number of ticks is 10.

242	0.2	GENERAL	The user shall be warned that if they leave the simulation screen then all unsaved data will be lost.
301	0.3	UI	The simulation screen shall display an inactive results tab.
302	0.3	UI	The results screen shall activate the results tab.
303	0.3	UI	The results tab shall receive focus when the software advances to the results screen.
304	0.3	UI	The results screen shall retain the world view and the console view from the simulation screen.
305	0.3	FILE I/O	The software shall save events to a log as they occur. The log shall be named based on the start time of the simulation.
306	0.3	UI	The results tab shall display each active template, its original population size, its result population size, the number of times the template's instances reproduce, the average life length of the template's instances, and genetic drift. Genetic drift is defined as the change between the templates' instances' average trait values at the beginning and end of the simulation.
307	0.3	USE CASE	The results screen shall allow the user to save the results of the simulation,
308	0.3	USE CASE	The results screen shall allow the user to return to the configuration screen with all template data intact.
309	0.3	USE CASE	The results screen shall allow the user to view the entire events log from the simulation.
310	0.3	UI	The events log, upon user request, shall be displayed in a modal pop-up window with a text area and scroll bars.

6.2Non-Functional Requirements

Req. #	PR	Category	Description
1	0.0	NON-FUNCT	The software shall be packaged in a JAR file
2	0.0	NON-FUNCT	The software shall be tested on Windows 7, version 6.1 (build 7600) 32- and 64- bit.
3	0.0	NON-FUNCT	The software shall be tested on Windows Vista, version 6.0 Service Pack 2 (SP2) (Build 6002) 64-bit.
4	0.0	NON-FUNCT	The software shall be tested on Windows XP, version 5.1 Service Pack 3 (SP3) 32-bit
5	0.0	NON-FUNCT	The software shall be tested on MacOS 10.6 32- and 64-bit
6	0.0	NON-FUNCT	The software shall be tested on Ubuntu Linux, version Lucid Lynx (10.04) using Linux Kernel 2.6.32 64-bit.
7	0.0	NON-FUNCT	The Java platform for all testing shall be Sun Java, version 1.6.0.22 (or greater)
8	0.0	NON-FUNCT	The platform the program runs on shall have a minimum resolution of 800x600 pixels

9	0.0	NON-FUNCT	The software shall be licensed under the Artistic License version 2.0 in accordance with the SPMP.
10	0.0	NON-FUNCT	The software shall be expected to run on any OS and hardware platform (both 32-bit and 64-bit) that fully supports and uses Sun Java version 1.6.0.22 (or greater)
11	0.0	NON-FUNCT	Exceptions that can't be handled by the software are logged to a local error.log file.
12	0.0	NON-FUNCT	User input data shall be validated and the user presented with an error if they input invalid data.
13	0.0	NON-FUNCT	Local exception handlers shall be included for all file handling conditions and the exception will be propagated to the user in some form.
14	0.0	NON-FUNCT	The system real-time clock shall trigger a tick no sooner than every second (or when the previous tick is completed) by default.
15	0.0	NON-FUNCT	For all output files, the software shall require a minimum of 5GB available storage space.
16	0.0	NON-FUNCT	Performance shall scale according to system resources (available memory, processor, etc)
17	0.0	NON-FUNCT	Additional requirements not covered by the functional specification are considered "gold-plated" and will be addressed in a new revision.

7. Post Mortem

7.1 Deliverable Effort

Estimated Hours	Actual Hours
30	25
40	38
40	57
40	62
60	62
40	
30	
10	
290	244

7.2 Individual Contribution

Item	Dale Earnest	Mike McWilliams	Jeff Dunn	Dong Luo	Total Hrs.
Meetings	6	6	6	6	24
Communications Plan	0	0	0	0	0
Application Investigation	3	6	3	4	16
Prototype (coding)	0	0	0	3	3
Prototype (drawings)	0	1	0	0	1
Project Proposal	0	0	0	0	0
Document Editing	8	2	3	1	14
Document Review	1	1	1	1	4
Total Hours	18	16	13	15	62

7.3 Team Effectiveness

7.3.1Team Meeting

Score: 10

Details: Meetings are effective, on topic, with little distraction. Team members are focused, contributing, and eager to complete tasks on the agenda.

7.3.2Document Preparation

Score: 7

Details: Communication is good but timeliness needs improvement. Real life time factors hurt document preparation with an impact on quality.

7.3.3Development

Score: 5

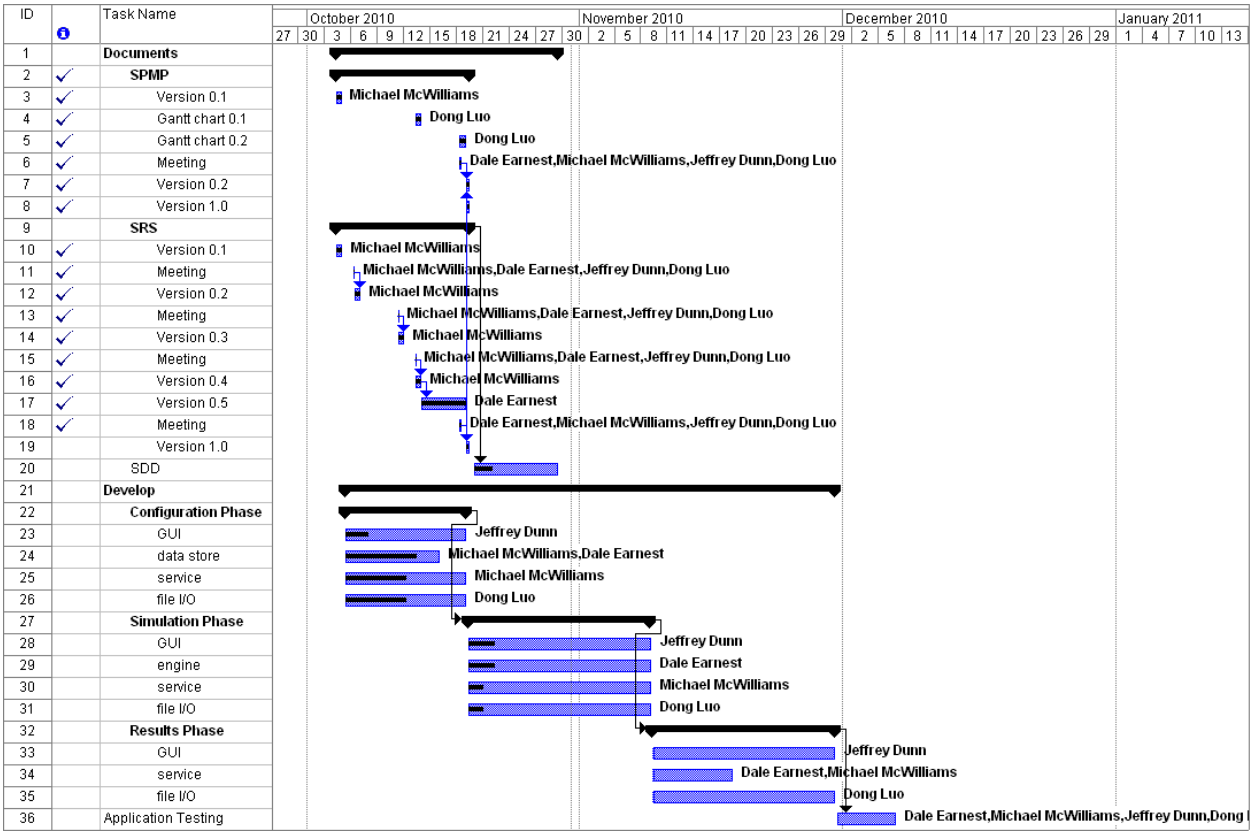
Details: Timeliness needs improvement; currently behind schedule. Code quality is good and communication between team members is high, but we're behind schedule due to real life time factors.

7.3.4Summary

Score: 22 out of 30

Details: This score indicates that the project is in a warning zone, that though we have some strong team members we are in danger of falling too far behind in development to catch up.

7.4Gantt Chart



Milestones Accomplished

- Completed SRS.
- SDD is 20% complete (UML diagrams).
- Team Meetings have met on time.

Warnings

- Everyone is behind schedule in coding due to life commitments. This risk has not been retired as the project manager's wife has been traveling for work and he's been doing single dad duty, cutting down on his time to assist in coordinating and finalizing documents. We will have to make up that time before the next meeting. We're looking to have the configuration phase done before the next class and that is our next milestone.