THE UNIVERSITY of NEW MEXICO

World Food Production and Land Management

Serious Game Project

Part 1

version 1.2

Background, Motivation and Project Goals

The past decade has seen a rising of reports on the future of the world's food supply. Many of these have emphasized the challenge of providing for a growing, increasingly wealthy population, or potential obstacles such as climate change. A growing number of projects throughout many countries are attempting to plan for a future that may be radically different than the present in difficult or impossible to predict ways. For example, some models predict that an increase in average global temperature could, via a slowdown of the thermohaline circulation, (one example of such circulation being the Gulf Stream) trigger localised cooling in the British Isles, France and the Nordic countries. However, the chances of this or any of a plethora of different scenarios occurring are unclear. It is currently observable fact the past few decades have shown significant and accelerating changes to the Earth's ice caps, its atmospheric composition, its average ocean temperature, average ocean salinity, land vegetation and other measurables. How far, and at what rate, those changes will continue in the future, and what will be the effects of those changes is not easy to predict. Additionally, the Earth's population of humans is undergoing significant and accelerating changes in size in movement from rural to urban, in industrialization and in many other ways.

Therefore, there is a great need to:

1) Improve the accuracy of global predictions.

2) Find effective interventions to large and sudden changes.

3) Find, plan and execute ways in which we humans can cope with or adapt to whatever changes do happen.

This project focuses one particular, yet very important, aspect of the third need: How can policy makers in the United States of America can best adapt strategies of food production to promote, on a global scale, freedom from hunger, freedom form sickness, security and the opportunity for prosperity for all?

To this end, the United States State Department has charged the University of New Mexico with developing a computer game where in a player plays the part of collective U.S. policy makers who can directly control food production strategies within the United States and also have some mechanism for influencing policies in other countries. For example, by U.S. policies that includes sharing of technology, aid, training, and/or international agreements.

This project is being worked on by a number of UNM professors, scientists form Sandia National Labs and a student build project that crosses the boundaries of different classrooms and different colleges within UNM.

This game will incorporate existing, but stat-of-the-art models of global prediction of factors likely to affect food production, food distribution, food consumption, and food safety.

The goal of this interdisciplinary cooperative effort by the end of the spring 2014 semester is to have a working, playable, fun, proof-of-concept game.

What is meant by a "proof-of-concept" is that:

1) Many of models will not be complete and some parts of the model may be much less complete than others.

2) Much of the data may not be complete, for example, the game may model some regions with plant hardiness zone data values on a resolution of 100 square miles while other arable land areas may be modeled at a much courser scale. For a point of reference, 100 square miles is approximately the size of a small US county along the eastern coast. Bernalillo County, for example, is 1,169 square miles (3,028 km^2). Indeed, if a variable grid size is to be used, it would likely make more sense vary grid resolution by variability in soil, altitude, water, and other parameters relevant to food production. However, in the proof-of-concept, we may need to tolerate some highly arable and varied land areas being modeled as homogeneous.

3) Many of the player control features may not be implemented.

4) Running the proof-of-concept game may require a higher end desktop than the target machine.

5) The proof-of-concept requires the model to be accurate enough to prove the concept: that is, it needs to demonstrate to professional in the field, that the models are working correctly with the limited data given.

6) The proof-of-concept requires that the game be playable and fun – at least for a short time. If, for example, we expect that the full game will be rich and variable enough that players can enjoy for 100 hours of play, then the proof-of-concept needs to be just as fun as the full version, but maybe after 30 minutes, the player has tried everything an seen all there is to see.

7) The proof-of-concept may contain some "bugs". For example, the models used may fail under some special cases or the user interface may occasionally lock up. By "some" what is meant is a level far more than would be tolerated in a released product, but not so common that play-testers stop thinking the game is fun or stop thinking the final game can ever teach anyone anything.

User Stories

1) The player is expected to be an average high school student, college student or adult who is not a professional in the field.

2) More than 75% of players should find the game fun. At least 10% should really love it and 90% should not find it annoying.

3) Players should feel like they are learning something of what choices U.S. policy should make.

4) Professionals in the field, watching a player, should feel that the player is actually learning something about U.S. policy making and NOT learning false or hollow lessons.

5) Some players, who happen to really like the game and play it for many hours, can go beyond just learning known concepts, and actually generate knowledge. That is, the game can help some players think of solutions that no one has thought of before and professionals regard as good ideas.

6) The game (at least the proof-of-concept) will be single player.

7) The game will be played on a display that has an inside diagonal of at least 15.6 inches and has at least 2 megapixels.

8) The game will be free and open source.

9) The time scale of the game is from the present through 2050.

10) The player could start the game at the present and play through 2050 or could select from some pre build scenarios that start with a troubled Earth sometime in the future and the player's task is to figure out what is wrong and fix it.

11) When the player starts the game, all of the variables of all areas should default to their actual current values (or to some scenario values).

12) The models must be stochastic. That is, if a player makes the same choices two games in a row, the results should not, in general, be the same. Yet, the models must also have desirable chains of cause and effect so that 1) as a player becomes more experienced with the game, the player steadily improves his or her ability to produce desired outcomes, and 2) professionals in the field agree that the cause and effect chains represent a wide range of reasonable start-of-the-art believes. That is, the game should teach the player accurate and unbiased (or at least polybiased) information.

13) If the player leaves all defaults unchanged and clicks play, then by 2050 there must be massive human deaths, with starvation, wars, birth defects and a return to a stone age. Even if this is not what would really happen if we just continue as we are currently, we are making a game and it would be silly if the winning strategy is to not play.

THE UNIVERSITY of NEW MEXICO

Requirements for the First Milestone: First Pass at an Interface

1) Milestone 1 is due on Tuesday, Feb 10 at 12:30 PM (at the start of class) in Blackboard learn. The full project source code and all parts needed to compile and run must be in .zip archive and submitted in accordance with the submission policy detailed in the CS-351 course syllabus. Also, on that Tuesday, each group will give a 2 to 3 minute demo of their running work.
2) All source code will be written in Java 1.7, will compile and run with Oracle's Java 1.7 JDK.
3) Graphical User Interface components will use some combination of The Java Foundation Class Swing and/or the Lightweight Java Game Library (LWJGL).
4) The program will start up showing a world map. This may be one of the 2D projections or maybe a 3D render (using LWJGL).
5) XML parsing will be done using Oracle's SAX library.
6) Beyond the SAX reader and LWJGL, some groups may want to use other third party packages. Before using any such package, each group must receive explicit permission from the CS-351 course instructor (Joel). As the software developed as part of this project will be will be made open-source, all third party packages must be freely distributable (although third party packages need not be open-source).
7) Any third party images, sound or other resources must be 1) clearly sited and 2) licensed for free redistribution with or without modifications to those resources.
8) All student developed source code will meet the CS-351 coding standards (see course website for details).
9) All classes and all ***public*** methods must include JavaDoc. You can use an auto generator for creating place holders, but it will be obvious if you do not edit the auto generated text. Class JavaDoc must explain how the class is designed to be used. Method JavaDocs must list and explain input, return values and any side effects. You may also sometimes choose to include explanations of how the method does what it does if you feel it will help your classmates who may be editing your code in milestone 2 or 3.
10) During development, all groups will ***correctly*** use a non-publicly viewable Git repository (such as GitHub) for version control. Correct use of a Git repository for a project this size means:
    a. Having ONLY ONE BRANCH.
    b. Both team members need to be making frequent commits to that one branch.
    c. Only working code should be committed to the repository.
    d. Each commit must include a one or two line comment describing the changes.

e. Commits must be frequent enough so that a line of text is sufficient to describe the changes made.

In addition to the two team members, read access to the repository must be given to the CS-351 course instructor (GitHub account castellanos70) and lab instructor (GitHub account TBA).

11) This first milestone will be programed in teams of two. Each team will be following the same set of requirements, but there is significant variation in how the end products will run.

12) The code ***must*** be very modular. At the start of the second milestone (which will be groups of 3 and many groups with totally changed members) , I want to be able to start every group out with identical copies of 2, 3 or 4 versions of milestone I code. Also, I want those versions to consist of the best parts of all the groups' code. For example, one group may have a great zoom mechanic, but bad error reporting in the xml and a third group may have the best selection method, I want to create versions that contain the best of different projects. This means making **very** modular code.

13) Every group's base directory (which must be included in the .zip archive) must be: **CS-351_Pro1_M1_**`firstName1_firstName2`.

14) When the program starts, it must read all .xml files within the directory:

<projectroot>/resources/areas/

Each configuration file will be a series of <area> tags that is an ordered list of the area's the vertices. The last vertex is assumed to be connected to the first. For example

```
<area>
<!-- The area below happens to mostly follow the actual -->
<!-- political boundaries, but my guess is that the game-->
<!-- configuration files will be mostly rectangular.    -->
  <name>Bernalillo County</name>
  <vertex lat="35.214910" lon="-106.244921"/>
  <vertex lat="34.953351" lon="-106.244234"/>
  <vertex lat="34.952788" lon="-106.150850"/>
  <vertex lat="34.870580" lon="-106.410402"/>
  <vertex lat="34.889168" lon="-106.411776"/>
  <vertex lat="34.902685" lon="-106.427568"/>
  <vertex lat="34.905500" lon="-106.685060"/>
  <vertex lat="34.869453" lon="-106.720766"/>
  <vertex lat="34.870016" lon="-107.024950"/>
  <vertex lat="35.219678" lon="-107.196611"/>
</area>
```

The intent is not for an area named "Bernalillo County" to necessary follow the actual political boundaries of Bernalillo County. Rather, it would make sense to have it follow

geographical features, such as a river valley, and name the area for whatever political or place name it happens to include.

Within the game, each "area" would be considered homogeneous: that is, the same planting zone, the same temperature, the same elevation, same rain fall, etc. Crops within an area would be specified as, for example, 20% corn, 10% beans, 50% suburban, 15% dry grasses, 5% spars juniper forest. There is no way to specify the layout of these ground covers within an area. Note that none of this data would be specified within the area files. Rather, each value, such as average rain fall, goes in its own configure and is mapped to the area files using the area name.

There is no specified maximum to the number of areas. The total land area of the Earth is 57,500,000 square miles. Thus, if the whole land surface were divided into square areas that are ten mile on a side, then the land would have 575,000 areas. Do not hard code arrays to this size, but in thinking about data structures and access methods, think of this as a likely upper limit.

For milestone I, the other classes working with us will not yet have this data. Therefore, you should make up your own files for testing. The simplest would be procedurally generating random names and dividing the land masses into equally sized rectangular areas – not worrying if some areas mostly overlap large lakes and completely disregard abrupt biome changes.

15) If a configuration file contains malformed or impossible data, then the reader must:

    a. Display a scrollable window that shows the offending data file.
    b. Highlight the offending line and sets the scroll bars so that the line is visible.
    c. Provide a helpful error message.
    d. Allow the user to edit the data file and resave it.

    Impossible data includes:

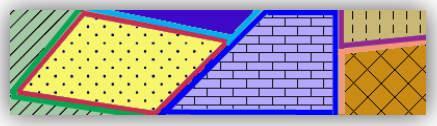
    a. Latitude value greater than +90 or less than -90.
    b. Longitude value greater than +180 or less than -180.
    c. Polygonal region with less than three vertices.
    d. Polygonal region with intersecting edges.
    e. Polygonal region that is degenerate (the vertices are collinear).

16) There must be some way for the player to select a location on the starting map and ***smoothly zoom*** in on that region. The player must not be able to zoom out past the starting view of the whole Earth. The player needs to be able to zoom in so that the smallest named area in view is large enough to display its name in a 12 point font where all or most of the name is within the area. Keep in mind that some areas may contain much larger named areas than others. Exactly, this is to be handled, is up to each team.

17) There must be some way for the player to easily select multiple adjacent areas to enable the player change game variables at once for all selected areas.

18) The player must be able to select from twelve different types of data to display:
    a. planting zone,
    b. Percentage of each type of crop, or other ground cover (forest, lake, urban, ...) For this milestone, it is for you to decide how many types of crops and what they are. In the final milestone there will likely be many types across the world, but only a few types in each area. Therefore, an area should not display any information about crop types have zero percentage presents in that area.
    c. annual rainfall,
    d. monthly rainfall,
    e. average monthly high temperature,
    f. average monthly low temperature,
    g. population,
    h. cost of crops,
    i. profit from crops,
    j. elevation,
    k. happiness of population,
    l. soil type.

    Some ways that multiple values can be displayed at once might be to use one parameter to set the outline color of the area, another parameter could be used to set the color of the interior of the area. Which parameter is used for each of these could be user selected or fixed – your choice. A mouse-over on the area could display a small box listing crop percentages and/or other data. Area colors could be overlayed textured patterns that indicate a different parameter. For example:



    This needs to be flexible. I, not knowing much about food production, made up this list. The list that the other classes actually want us to display will likely be very different. They have not yet even decided on the number of parameters they want to show which may be more or less than the 12 that I picked.

    Your GUI needs to work, that is, when I select different displays, I should see different values displayed. However, it is not your job to get the data. Therefore, for testing, generate random data – maybe warmer at the equator and such, but largely random.

19) Player must be able to select between English and Metric units.

20) Different groups may experiment with different input devices: one may be more keyboard based, another more mouse based. Still another may want to use LWJGL to

interface to a game controller or a Wiimote (I have extras of these with USB blue tooth receivers that will work in almost any laptop or desktop).

21) Your program must have time: some way to start it, stop it and change its speed. When your program runs, it should call placeholder functions that, for milestone 1, make random changes to the each of the values that are displayable. Note that "random changes" is different from "random values". Random changes means that each time step, the current value is increased or decreased by some amount.

That is a lot of specification, but it is also a lot of freedom. Use some creativity. Try out some experimental ideas. Maybe it will get picked as best or maybe it will spark a new idea when mixed with some other team's experimental idea.

Grading Rubric

coming soon...