Ski Resort Simulator

Group #10



Software Engineering

February 17, 2012

http://www.skiresortsimulator.com

Matthew Moccaro

Nancy Andia

Eric Wengrowski

Erjon Malaj

Walid Al Jabowbi

Jae Lee

Table of Contents

1. Individual Contributions Breakdown 3
2. Customer Statement of Requirements 4
   a. Problem Statement 4
   b. Glossary of Terms 6
3. System Requirements 7
   a. Enumerated Functional Requirements 7
   b. Enumerated Nonfunctional Requirements 8
   c. On-screen Requirements 10
4. Functional Requirements Specifications 12
   a. Stakeholders, Actors & Goals 12
   b. Use Cases 14
      i. Casual Descriptions 14
      ii. Fully-Dressed Descriptions 14
      iii. Use Case Diagram 18
   c. Traceability Links 19
   d. System Sequence Diagram 20
5. User Interface Specifications 22
   a. Preliminary Design 22
   b. User Effort Estimation 25
6. Domain Analysis 27
   a. Domain Model Concepts 27
      i. Domain Model Diagram 28
      ii. Associations & Attributes 29
      ii. Traceability Matrix 31
   b. System Operation Contracts 32
   c. Mathematical Model 34
7 Plan of Work 38
8. References 39

Individual Contributions Breakdown

	Eric Wengrowski	Erjon Malaj	Jae Lee	Matthew Moccaro	Nancy Andia	Walid Aljabowbi
Project Management (10 points)		75%			25%	
Customer Statement of Requirements (9 points)		20%			80%	
System Requirements (6 points)				50%	50%	
Functional Requirements (30 points)		20%	30%		10%	40%
User Interface Design (15 points)				100%		
Domain Analysis (25 points)	65%	15%	10%	10%		
Plan of Work (5 points)		30%	35%		10%	25%

Customer Statement of Requirements

Problem Statement

A national survey shows that the number of skiers in the Northeast was of approximately 13.4 million in 2010; with this high number of skier visiting resorts, the experience of visiting a ski resort might not always be a pleasant one. When the resort is not crowded, the skier will have more space to enjoy him or herself. However, if the resort is crowded, the skier should stay home to avoid an unpleasant experience and waste money, and visit the resort on another day when it is less crowded. This is the main reason why skiers are actively looking for less crowded days to attend ski resorts and have a positive experience. The goal of this project is to create a simulation that will allow users to choose the values of different factors, in order to help them decide whether it will be a good idea to attend the ski resort on a particular week or not, as well as to analyze how each factor affects overall success rates. The simulation will reflect the minority game, where if the minority or less than 60 percent of players attend, then the resort won't be crowded and those who went consider it a success; however if the majority of players decide to go, then the resort will be crowded and the player will have a negative experience during their visit.

The ski resort program is designed to allow players to learn whether or not to attend a ski resort based on different factors which can influence an individual's decision. The factors are: game percentage, the weather in degrees (better weather entices more people to go), skill level which ranges from 1-10 (the higher the skill the more often the person will want to go), number of friends, since a person would be most likely to go if their friends are also going, and their own experience during the last trip. Other parameters that individuals can set are the number of players used in the simulation, and how long the simulation runs for. The agent can also input what percent of the decision whether or not to go is based on strategy, and what percent is based on these external factors like weather, friends, etc. Based on these factors, the simulation will produce a set of graphs that the individual can analyze. For example, the individual may notice that as the weather gets worse, it is likely that going to the ski resort results in a positive experience, or that as a greater percentage of the decision

is strategy, the higher one's chance of having a positive experience. Using information like this, the individual will be better educated about how to make a decision about whether to go.

This program is also intended for ski resorts themselves who are interested in raising profits and boosting attendance. After choosing this mode in the beginning, instead of the individual mode, the ski resort can set the same parameters as an individual, as well as a lower bound percentage, below which they don't turn a profit. The idea is similar in that they don't want too many people to go, because they will have a negative experience and may not go next time, but they also want a minimum number to go to keep the resort open. After running the simulation, the ski resort can analyze graphs that may convince them to change their marketing strategy. For example, maybe larger groups result in low success rates so they may offer deals for only 3 or 4 instead of 8. Maybe having all skilled skiers also results in lower success rates so they may want to target individuals they may have overlooked by offering deals for first-timers.

Of course, all of these predictions are only estimated guesses, but they illustrate the usefulness of a program like this to both individuals and ski resorts. That coupled with an intuitive user-interface makes this program a valuable resource.

Glossary of Terms

Agent - The player who will choose the different factors to start a simulation.

Decision Percentage - Percent of the decision-making that is strategy.

Game Percentage - Percent that determines if the week was a success or not, for example if more than this percentage go, it is considered an unsuccessful week for those that went, and successful for those that didn't.

Lower Bound - Percent below which the ski resort considers the week a failure (no profit).

Reset - The simulation will reset everything to default values and clear the graphs.

Simulator - The Graphical User-Interface which allows the players to participate.

Strategies - The combination of factors chosen by each player for each simulation.

Success Rate – Whether for the ski resort or the individual skier, this is the number of successes (attending when not crowded or didn't attend when crowded) divided by the number of weeks that have went by

User - Referred to as this before choosing Individual or Ski Resort in the beginning.

Weather Preference – Defined as the coldest temperature that the skier will endure. For example if one's preference is at least 40 degrees and the temperature is 50 degrees, this factor in the decision equation will be a 1, enticing the skier to attend

System Requirements

Enumerated Functional Requirements

Identifier	Priority Weight	Requirements
REQ1	4	The simulator will allow users to choose whether they are an individual or a ski resort
REQ2	3	The Game Percentage, Number of Agents, Length of Simulation, Weather/Temperature, and Average Number of Friends can be set by both Individuals and Ski Resorts
REQ3	2	The Experience Percentage and Skill Level can be set only by a ski resort
REQ4	2	The Decision Percent can be set only by an Individual
REQ5	3	Upon Completion, the simulator will produce a few lines of relevant output statistics and allow the user to click a graph
REQ6	2	The graph Average Number of Friends vs. Success Rates, Game Percentage vs. Success Rates and Weather vs. Success Rates will be viewable by both Individuals and Ski Resorts
REQ7	1	The graph Decision Percent vs. Success Rates and IQ vs. Success Rates will be viewable by Individuals only
REQ8	1	The graph Skill Level vs. Success Rates and Experience vs. Success Rate will be viewable by Ski Resorts only
REQ9	3	The simulator will allow the user to complete multiple runs and update the graphs for each run
REQ10	5	The simulator will store data in memory for each run, and allow the user to terminate the program, restart and continue
REQ11	5	The program will allow the user to run the simulation, but it will run only run if all textfields are correctly input/formatted.
REQ12	3	The program will allow the user to save their state (individuals and runs) to a file, close the program, open it later and continue. (Persistent data storage)

Enumerated Non-Functional Requirements

These are ordered by priority for our particular program.

REQ-13: Usability: The human factor in this simulator has been kept to a minimum to ensure ease of use. The user will simply need to adjust settings and click the begin simulation button. This minimizes the amount of mouse clicks and keystrokes needed to operate the simulator. Also, preset settings are there to guide the user if need be. Aesthetics will include a colorful and graphic GUI which will be visually appealing with large easily clickable objects such as buttons and sliders. Consistency will also be kept as well considering that there will only be two objects, sliders and buttons, to input all of the parameters of the simulation. This will keep things easy to use and consistent so the user will know what is coming next. Finally, documentation will include a help file which will explain in step by step methods how to use every aspect of the simulator. This is our most important requirement as our program should function very well due to its inherent simplicity.

REQ-14: Functionality: This requirement is expected to be extremely easy to meet. Therefore, it is a high priority because, if it does not work, it will defeat the purpose of the entire program. The feature set will include many things. The first thing it will contain is the ability to simulate how many people will attend a ski resort over several different time periods with different factors affecting attendance. This will include factors such as weather and intelligence of each member who is considering attending the resort. Another thing it will contain is preset scenarios in which the user will have some guidelines to begin simulating easily and will be able to create custom settings as they become more experienced with the program. The capabilities of our program include the ability to model many real world aspects of our attendees, such as intelligence, and things such as if their friends are going as well. It allows the user of the program to choose how many people will be possible future attendees. For generality, the program will basically allow those who look to open such a resort to better plan for maximum attendance and profit. Security will be maintained because the simulation results will not contain any personal data.

REQ-15: Performance: The program being discussed should also have very high performance. This is an important requirement as the users will not want to wait 10 minutes for the simulation to run. While it only needs to run a single algorithm, even with all of the parameters on max, the calculations should be short enough to keep run time to a bare minimum. This should put our response time at probably a maximum of only a few seconds.

Efficiency and resource consumption should also be great for our program as the amount of random access memory used will be very low and actually mostly filled by our GUI. The throughput of our program will mostly depend on the speed of the processor on the machine it is being run. It is mainly mathematical calculations, and the faster these are done, the faster our program will print its results.

REQ-16: Reliability: Frequency of failure should be very low, therefore this is not as high of a requirement. Recoverability should also be very high, since restarting the application should easily reset the program after a crash. Predictability should also be high, as changing the sliders will also change the implemented graphs and output. Accuracy should be very high as well, considering we have finite values which include bounds to all of our inputs. Finally, the mean time to failure should be very long as it seems very unlikely that this program will fail unless a major bug is discovered.

REQ-17: Supportability: Our program will have much support in becoming a final product. This is not a high requirement however because it should be a minimal task to support this program. Testability will be high as we can test a wide range of inputs for our program, ensuring that every combination works well and does not crash our program. Extensibility will be low however, as our program should not need any extension beyond our final project. Adaptability will be the same as the program will not need to make any adaptations except to the inputs given by the user. Maintainability should be very good because the program will not take up much memory and should run on all machines considering that it will run on the JVM. This gives it great compatibility, localizability, and portability as well. Configurability is a large aspect of our program only for input parameters. We do not want the user to configure inner parts of the program as this may cause a crash so this will not be available. Serviceability will be low as this program should not need any service whatsoever due to its simplistic nature. Compatibility will be great, as we can use an install shield, which is commonly used for almost every program, making installing it an easy procedure.

On Screen Requirements

The following pictures illustrate the different forms with each of their commands listed and prioritized by number.

Opening Form:



The requirements for this form are:

1.) Simplicity – The user will be able to easily choose which option they would like to start with.

2.) Aesthetics – The form is the first thing people will see when they open our program. Therefore, it should be a great representation of our work

Ski Resort Form:



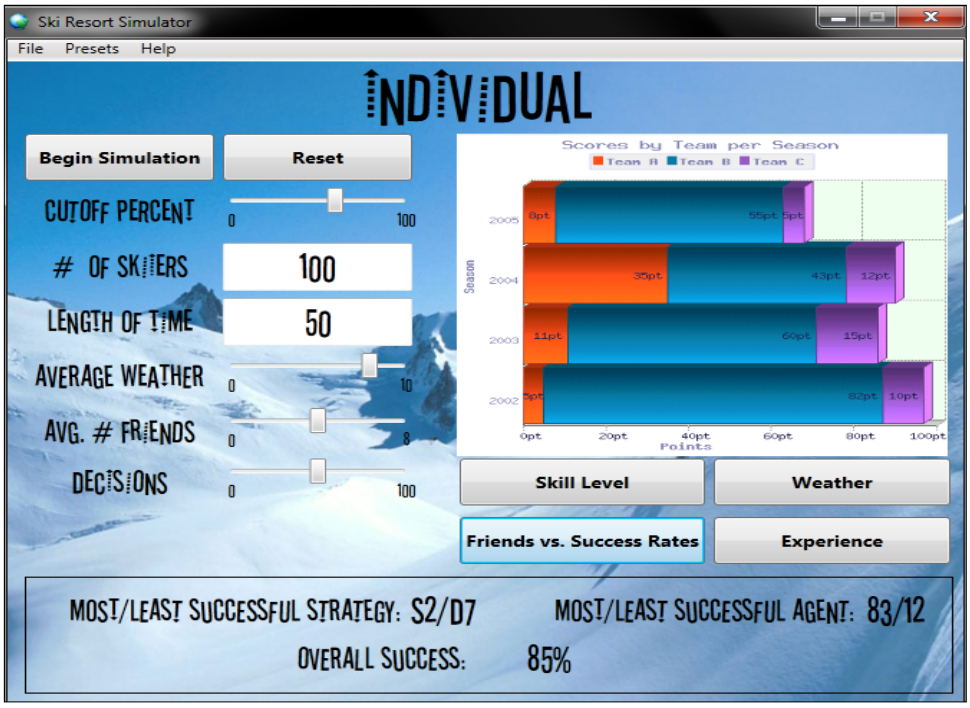
The requirements for this form are:

1.) Simplicity – The user should be able to easily change settings and options intuitively. This is met with sliders and buttons, simply to use interface items.

2.) Functionality – The program must run when the "Begin Simulation" button is pressed. It must also reset when necessary. The graphs must change easily as well when the graph buttons are pressed.

3.) Aesthetics – This form is something which people may spend a large amount of time viewing. Therefore, it should look as nice as possible for our users.

Individual Form:



The requirements for this form are the same as the "Ski Resort" form and prioritized the same way:

1.) Simplicity – The user should be able to easily change settings and options intuitively. This is met with sliders and buttons, simply to use interface items.

2.) Functionality – The program must run when the "Begin Simulation" button is pressed. It must also reset when necessary. The graphs must change easily as well when the graph buttons are pressed.

3.) Aesthetics – This form is something which people may spend a large amount of time viewing. Therefore, it should look as nice as possible for our users.

Functional Requirements Specification

Stakeholders, Actors and Goals

Stakeholders:

1. Individual Skier
2. Ski Resort/Investor

An environment like the ski resort can be used to establish two goals: individual enjoyment and profit. A skier/snowboarder main interest is to enjoy their time and overall have a good time. In contrasts, actors such as the investor and the ski resort have interests that are more economic based than self-interest. To simplify the problem and to reduce unnecessarily constraints, we will make the following assumptions: 1. Ski resorts and the investors always look for a way to increase profit even it results in lowering their customers' overall satisfaction. (Therefore, ski resorts and investors will only increase customer's satisfaction if profit is directly related to satisfaction) 2. Skier/Snowboarder will, in counterpart, always look for a way to boost their overall experience at the ski resort. (Eliminates any unnecessarily strategies that do not play a key role. ie: A snowboarder selling candy bars at the main lounge with main interest for profit.)

A ski resort can use this software in order to gain access to useful data and plan for possible expansion of the resort. For example, a temperature vs. number of attendance graph can be used to predict when they should staff more or less people. By properly matching the staff number with the number of people, expenses can be reduced and the overall satisfaction of the customer will increase. Moreover, if data points toward an increase in populations, the ski resort can plan their park expansion ahead of time. This can increase profit because it will increase the number of the customers a resort can service. Investors (Gift Store, Food Concessions, and Equipment Rentals) will benefit like-wise because it will allow them to introduce price-estimate, reducing unnecessarily loss due to supply exceeding demands and vice-versa.

It is hard to imagine why customer will benefit from the software. In a way, however, they are the ones who benefit the most out of this software. A customer without access to this software will have to compile a list of experience manually. He or she will risk having a bad experience at the ski resort because there is no way of

predicting how many customers will be at the resort in a given day. While this software does not guarantee a hundred percent assurance, the software will greatly increase the likelihood of having a great time at the resort.

Actor 1: Individual [UC 1[UC 3] [UC 4] [UC 5] [UC 7]
[Roles]: Provides data to the system and memory to be interpreted
[Type]: Initiating
[Goals]:
-To set number of agents, number of iterations, temperature, average number of friends, game percentage, decision percent
-initiate the program.

Actor 2: Ski Resort [UC 1] [UC 2] [UC 4] [UC 6] [UC 7]
[Roles]: Provides data to the system and memory to be interpreted
[Type]: Initiating
[Goals]:
-To set number of agents, number of iterations, temperature, average number of friends, game percent, experience percentage, lower bound percent
-initiate the program

Side: Memory/Hard-Drive [UC 1-7]
[Roles]: Provides data to the system and memory to be interpreted
[Type]: Participating

Use Cases

Casual Description

Some of the more simple, trivial use cases are UC-1: Startup, UC-2: SkiResortFields and UC-3: IndividualFields. A casual description of each of these is given below.

UC-1: Startup

The User runs the program and is prompted to toggle either Ski Resort or Individual. This represents REQ-1 and is a precursor to all other use cases, since this is the deciding factor for many variables and parameters when the program runs.

UC-2: SkiResortFields

If the user clicked "Ski Resort" during UC-1, then he/she is allowed to enter their preferences for the fields Game Percentage, Number of Agents, Length of Simulation, Weather/Temperature, Average Number of Friends, Experience Percentage and Lower Bound, representing REQ-2 and REQ-3.

UC-3: IndividualFields

If the user clicked "Individual" during UC-1, then he/she is allowed to enter their preferences for the fields Game Percentage, Number of Agents, Length of Simulation, Weather/Temperature, Average Number of Friends, Decision Percentage, representing REQ-2 and REQ-4.

Fully-Dressed Descriptions

Below are the fully-dressed descriptions for the more complex, critical use cases, which are:

UC-4: RunSimulation

UC-5: IndividualGraphs

UC-6: SkiResortGraphs

UC-7: SaveFile

Use Case UC-4	RunSimulation
Related Requirements:	REQ1-REQ4, REQ11
Initiating Actor:	Ski Resort or Individual
Actor's Goal:	To run an iteration of the program
Participating Actors:	none
Pre-conditions:	All data entries have been completed
Post-conditions:	Output is displayed and Graph buttons are enabled
Flow of Events	
->	1. User selects button "Run"
	Include::UC-1 to UC-3
<-	2. System checks all data entries for errors in GUI, then runs iteration Graph buttons are enabled
Flow of Events for Alternate Scenarios	
->	1. User enters invalid number in one of the fields (ex. 9 friends or temperature below 0 is entered)
<-	a. System detects error (9 friends would be interpreted as 8 and a temperature below 0 would be interpreted as 0, so entries are kept within bounds), System outputs error message
->	2. User edits field for which error was printed, clicks run (first run)
<-	a. System first initiates/creates the number of individuals that the user stated, then runs simulation
->	3. User edits a field and runs again
<-	a. System goes straight to run simulation

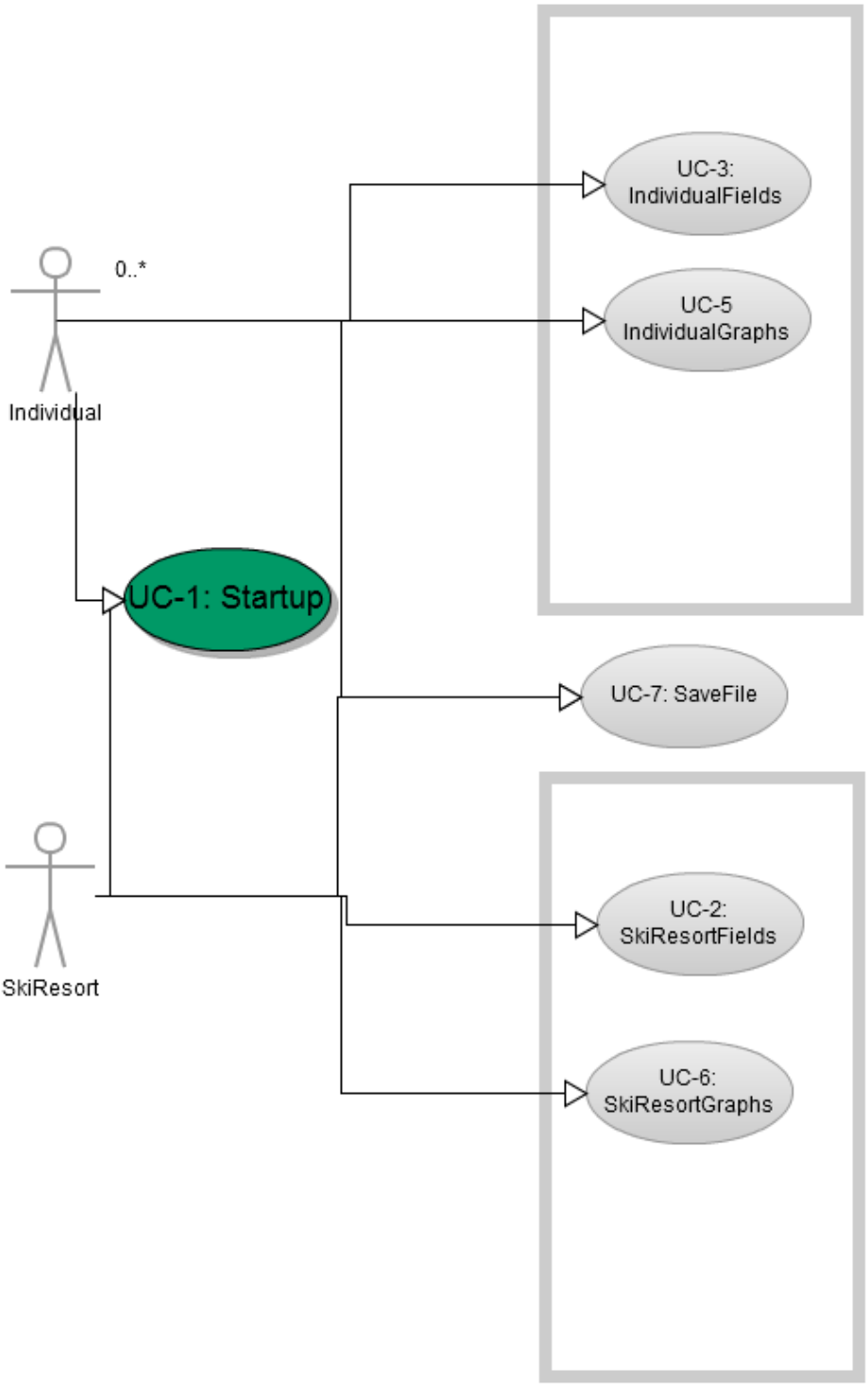
Use Case UC-5	IndividualGraphs
Related Requirements:	REQ5-REQ7
Initiating Actor:	Individual
Actor's Goal:	To open a specific graph
Participating Actors:	none
Pre-conditions:	Simulation has been run successfully with no errors/ incorrect inputs
Post-conditions:	Graph is displayed
Flow of Events	
	Include::UC-4
-> <-	1. User selects one of 5 buttons displaying graph of Success Rate vs. Average Number of Friends, Game Percentage, Weather Rating, Decision Percentage, or IQ 2. System displays appropriate graph

Use Case UC-6	SkiResortGraphs
Related Requirements:	REQ5, REQ6, REQ8
Initiating Actor:	Ski Resort
Actor's Goal:	To open a specific graph
Participating Actors:	none
Pre-conditions:	Simulation has been run successfully with no errors/ incorrect inputs
Post-conditions:	Graph is displayed
Flow of Events	
	Include::UC-4
-> <-	1. User selects one of 5 buttons displaying graph of Success Rate vs. Average Number of Friends, Game Percentage, Weather Rating, Experience or Skill Level 2. System displays appropriate graph

Use Case UC-7	SaveFile
Related Requirements:	REQ9, REQ10, REQ12
Initiating Actor:	Ski Resort or Individual
Actor's Goal:	To save state
Participating Actors:	none
Pre-conditions:	Simulation has been run successfully with no errors/incorrect inputs at least once
Post-conditions:	Present state is saved to file
Flow of Events	
	Include::UC-4
-> <-	1. User clicks "Save to File" and enters file name 2. System creates new file and saves state to file

Flow of Events for Alternate Scenarios	
-> <- ->	1. User clicks "Save to File" before having run the simulation a. System shows error message stating that at least one simulation needs to be run before saving 2. User runs simulation, then clicks save
<-	a. System saves state to file
->	1. User clicks "Open File" and chooses which file
<-	a. System loads previous state from file into memory
->	2. User edits a field and runs simulations
<-	a. System runs simulation
->	3. User clicks "Save"
<-	a. System updates file, saving new state

Use Case diagram



Traceability Matrix

	REQ Priority	UC-1	UC-2	UC-3	UC-4	UC-5	UC-6	UC-7
REQ1	5	X			X			
REQ2	4		X	X	X			
REQ3	3		X		X			
REQ4	3			X	X			
REQ5	2					X	X	
REQ6	1					X	X	
REQ7	1					X		
REQ8	1						X	
REQ9	3							X
REQ10	5							X
REQ11	5				X			
REQ12	3							X

Calculated Priorities for Use Cases (sum of Related Requirements' priorities)

UC-1: 5

UC-2: 7

UC-3: 7

UC-4: 16

UC-5: 4

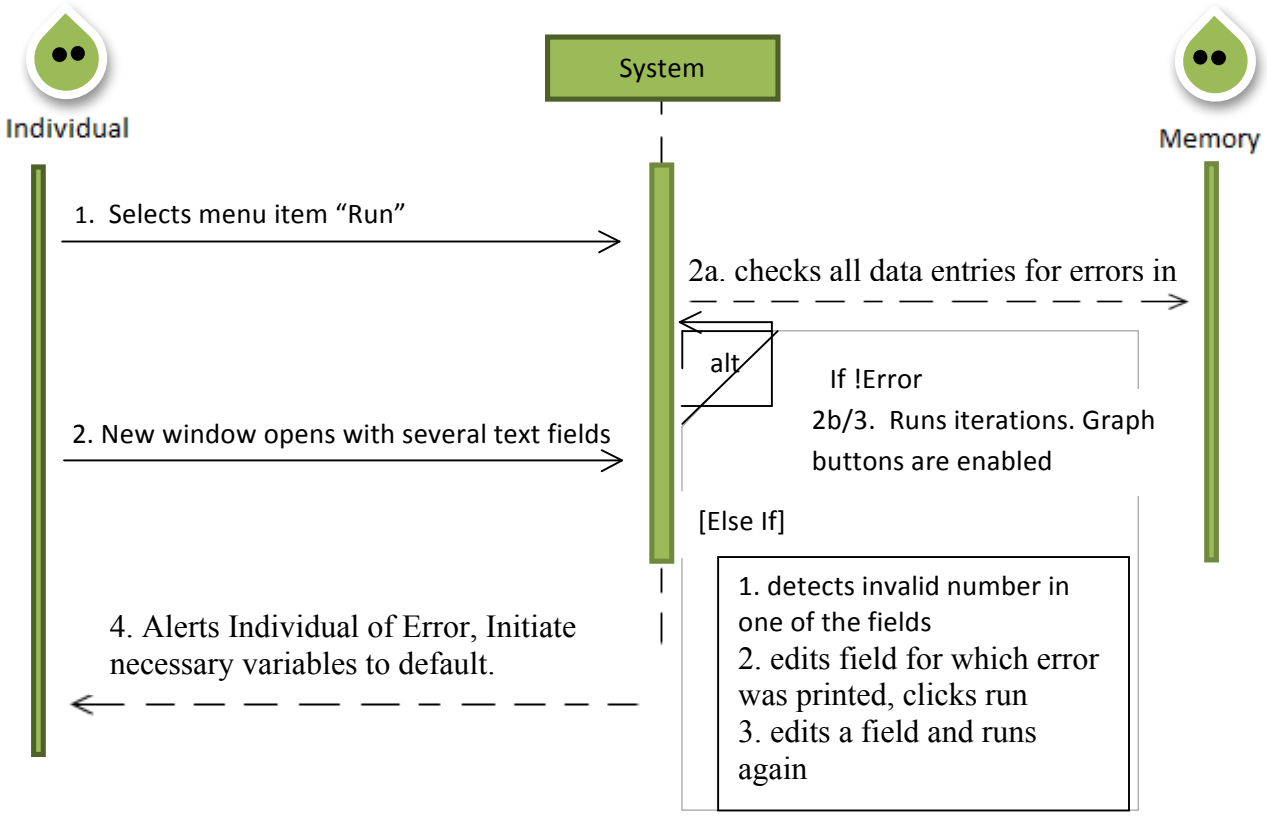
UC-6: 4

UC-7: 11

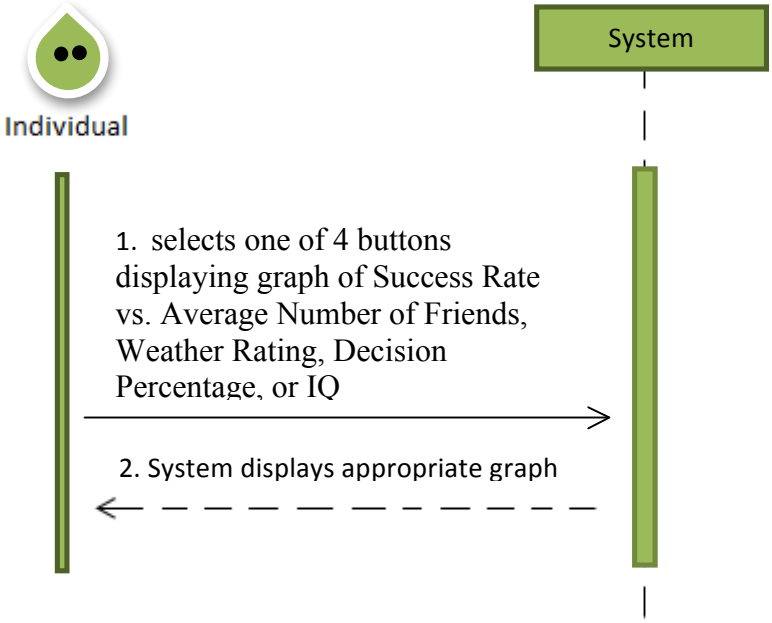
Using this metric, the Use Cases we will implement for Demo 1 are UC-1 (because it is an important parameter in the other use cases), UC-2, UC-3, UC-4 and UC-7. UC-5 and UC-6 (graphs) are more difficult and not as necessary for the purpose of our simulation because they are added features.

System Sequence Diagrams (for fully-dressed use cases)

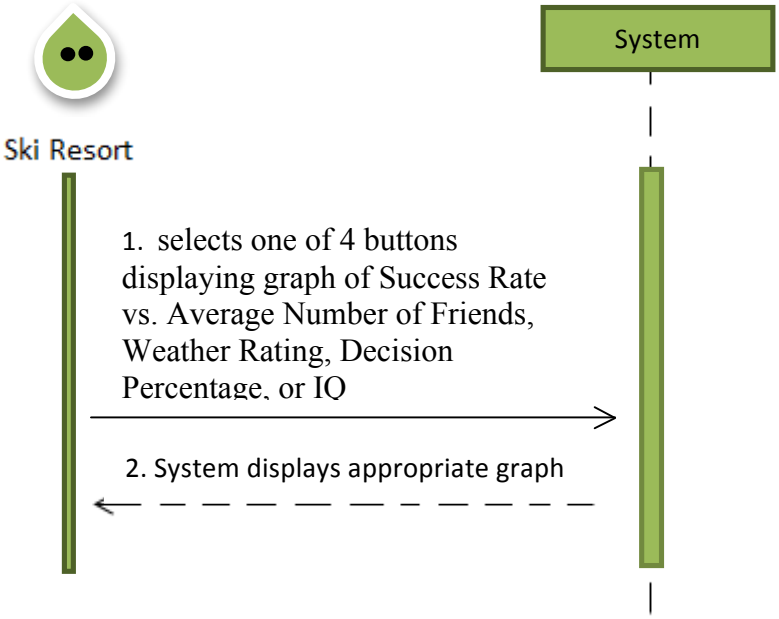
Use Case 4



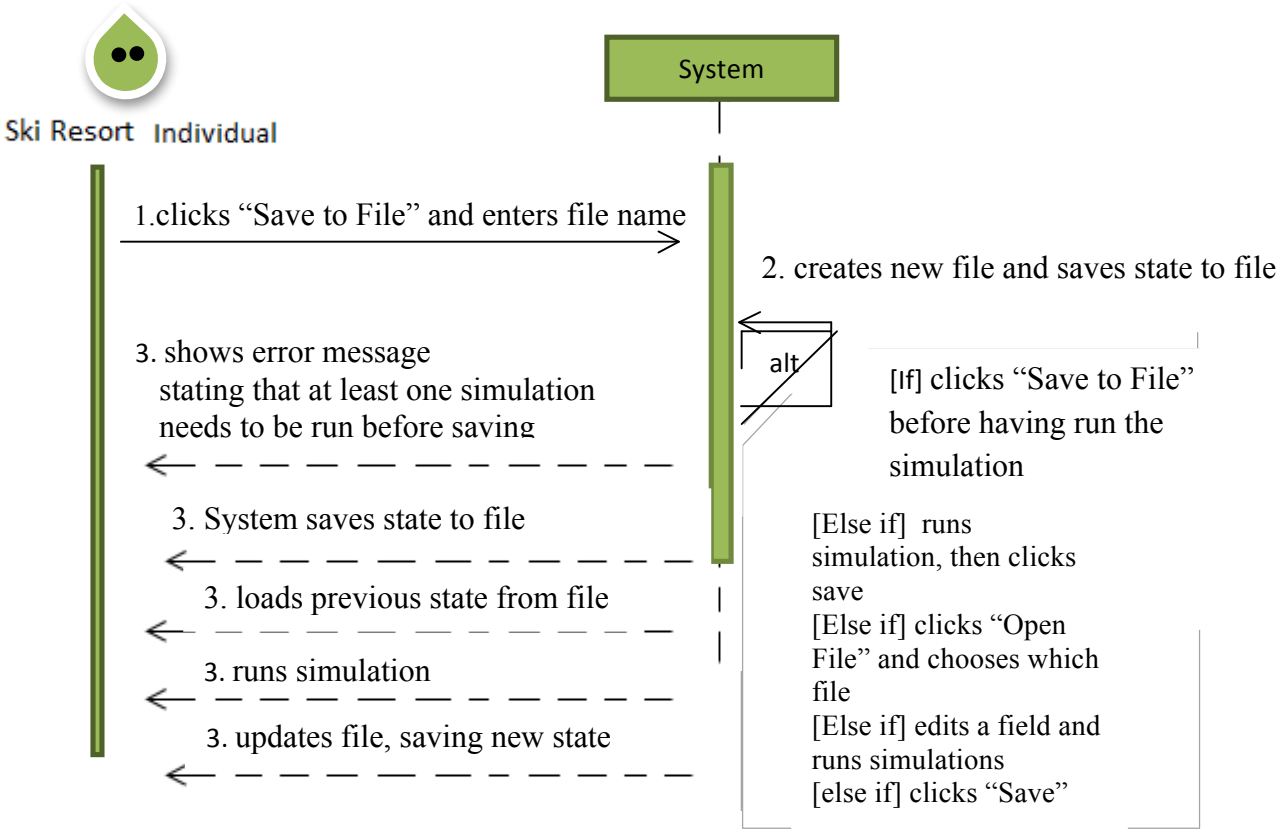
Use Case 5



Use Case 6



Use Case 7



User Interface Specifications

Preliminary Design

The navigational paths the user will follow when interacting with our application will be simple and efficient to ensure ease of use. The customer will begin by clicking on an icon to start the program. They will then be shown a simple window such as this:

Opening Window:



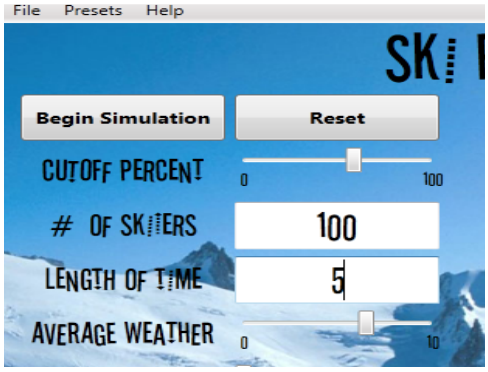
The user will then make a selection between either a Ski Resort simulation, or an individual's simulation, as shown in the window. The user will then be brought to their appropriate window. We will first take a look at the "Ski Resort" option. After clicking this button, the user will be brought to a screen such as this:

Main Window:



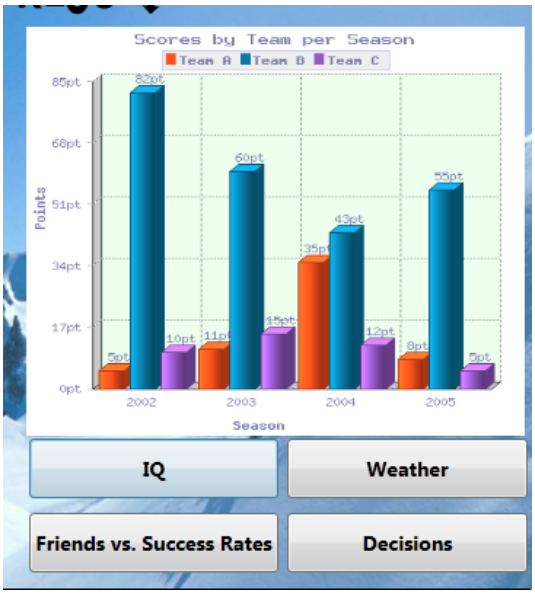
The user may then change all of the settings to their liking in on this form. This comes in a variety of two input forms, sliders and text boxes. These inputs fill the requirement of allowing the user to modify the scenario the simulation is run in to fit their needs. This is demonstrated in the picture below:

Sample Inputs:



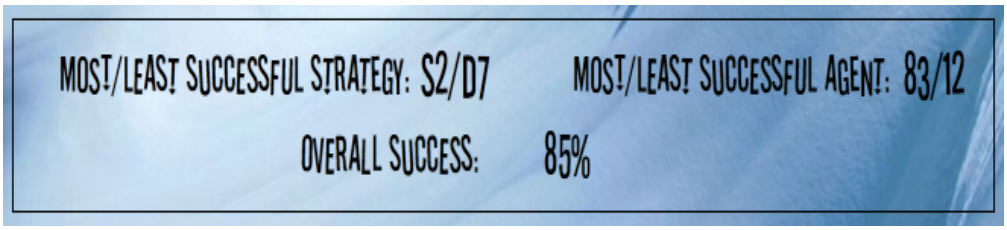
The user will then click the "Begin Simulation" button to start the process. They will then be shown one graph showing a certain output which fills the requirement of demonstrating how effective some factors are when attendance is concerned. They will be able to switch between this and other outputs by clicking the buttons below the graph as shown in the following picture:

Graph Display:



They will also be shown the most important data regarding the simulation in a box shown at the bottom of the window. This is shown here:

Sample Output:



Finally, the user can end the simulation and clear the graph area by selecting the reset button. This will then allow the user to exit, or to begin another simulation with the same, or different values. The user also has the option to use preset values in the menu at the top of the window. An example is demonstrated below:

Sample Graph:



Finally, the user may also have selected the "Individual" option from the opening screen. This is different only in its available parameters and a different set of graphs to display. An example of this UI is shown below:

Individual Graph:



User Effort Estimation

Usage Scenario 1:

1.) DATA ENTRY

a. Press the keys “1” and “0” and “0” in the “Number of Skiers” Box

b. Press the key “5” in the “Length of Time” Box

2.) NAVIGATION

a. Click the “Ski Resort” button.

b. Move the “Cutoff Percent” slider to 60

c. Move the “Average Weather” slider to 8

d. Move the “Average Number of Friends” slider to 2

e. Move the “Positive Experiences” slider to 25

f. Move the “Park Success” slider to 50

g. Click the “Begin Simulation” button.

h. Click the “Weather” button.

i. Click the “File” menu option.

j. Click the “Exit” menu option.

Number of Mouse Clicks: 10

Number of Keystrokes: 4

The fraction which goes to User-Interface Navigation is 10 out of 14.

Usage Scenario 2:

This is a worst case scenario. It includes all of the settings being adjusted and then reset.

1.) DATA ENTRY

a. Press the keys “9” and “5” and “0”in the “Number of Skiers” Box

b. Press the keys “1” and “0” in the “Length of Time” Box

2.) NAVIGATION

a. Click the “Individual” button.

b. Move the “Cutoff Percent” slider to 55

c. Move the “Average Weather” slider to 2

d. Move the “Average Number of Friends” slider to 4
e. Move the “Decisions” slider to 30
f. Click the “Begin Simulation” button.
g. Click the “Skill” button.
h. Click the “Friends” button.
i. Click the “Experiences” button.
j. Click the “Weather” button.
k. Click the “Reset” button.
l. Click the “File” menu option.
m. Click the “Exit” menu option.

Number of Mouse Clicks: 13

Number of Keystrokes: 5

The fraction which goes to User-Interface Navigation is 13 out of 18.

Usage Scenario 3:

1.) DATA ENTRY
    a. None
2.) NAVIGATION
    a. Click the “Ski Resort” button.
    b. Click the “Presets” menu option.
    c. Click the “Forever” sub-menu option.
    d. Click the “Begin Simulation” button.
    e. Click the “Weather” button.
    f. Click the “X” in the upper right hand corner.

Number of Mouse Clicks: 5

Number of Keystrokes: 0

The fraction which goes to User-Interface Navigation is 5 out of 5.

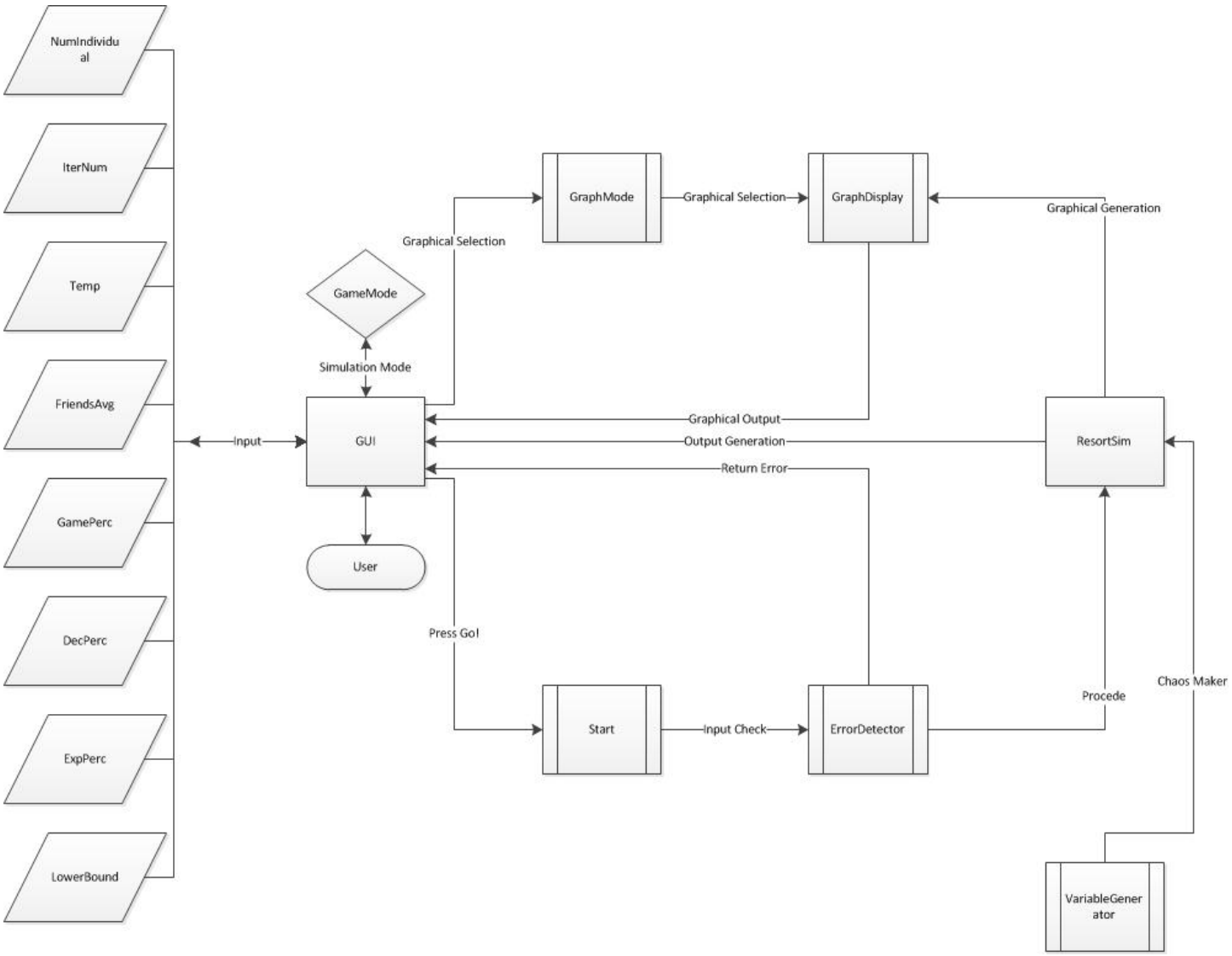
Domain Analysis

Domain Model

Concepts

Responsibility Description	Type	Concept Name
The container that remembers whether the user selects "Individual" or "Ski Resort" from the GUI menu.	K	GameMode
Container for the input for the number of agents involved in the simulation.	K	NumIndividual
Container for the number of iterations, comprising of weeks, months, or years.	K	IterNum
Container for temperature specifications.	K	Temp
Container for average number of friends (between 1-8)	K	FriendsAvg
Container for "Game Percent" percentage.	K	GamePerc
Container for user's input for "Decision Percentage."	K	DecPerc
Container for Ski Resort's input for "Experience Percentage."	K	ExpPerc
Ski Resort's input for Lower Bound Occupancy Percentage	K	LowerBound
The user determines which graph is to be displayed upon the GUI.	K	GraphMode
Generates pseudo-random variables for automatic simulations.	D	VariableGenerator
Initiates the ResortSim program.	D	Start
Determines if all values entered are valid. If they are not, an error message is displayed.	D	ErrorDetector
Generates graphs displaying the result of the simulation, to be displayed in the GUI.	D	GraphDisplay
Provides the user with a clear and simple way of setting game parameters and viewing simulation feedback and results.	D	GUI
Calculations are preformed with all assigned variables as the program simulates the Ski Resort minority game.	D	ResortSim

Domain Model Diagram



Associations

Concept Pair	Association Description	Association Name
GUI ó GameMode	The user determines whether the simulation will run in “Individual” or “Ski Resort” mode via the GUI. The GUI then allows the user to input the appropriate parameters.	Simulation Mode
NumIndividual ó GUI	The user inputs the number of agents involved in the simulation into the GUI.	Players
IterNum ó GUI	The user inputs the number of iterations to be preformed for the simulation.	Iterations
Temp ó GUI	The user inputs the temperature variable.	Temperature
FriendsAvg ó GUI	The user input the average number of friends for the agents in the simulation.	Popularity
GamePerc ó GUI	The user inputs the “Game Percentage” into the GUI.	Ambiguous Variable
DecPerc ó GUI	The user inputs the “Decision Percentage” into the GUI.	Decisions, Decisions
ExpPerc ó GUI	The user inputs the “Experience Percentage” into the GUI.	Skillz That Killz
LowerBound ó GUI	The user input the lower bound percentage of agents that will determine success rate into the GUI.	Going Along with the Crowd
GraphMode ó GUI	The user inputs the desired graph to be displayed.	Graphical Selection
GUI ó Start	The user initiates the start of the simulation after the variables have been input.	Press Go
Start ó ErrorDetector	The error detector function checks the see that all required variables have appropriate values.	Input Check
ErrorDetector ó ResortSim	ErrorDetector then returns a value that initiates ResortSim if all variables are valid.	Proceed
ErrorDetector ó GUI	ErrorDetector then returns a value that causes the GUI to return an error message to the user.	Return Error
ResortSim ó All Input Variables	All appropriate input variables are compiled by the ResortSim function.	Input
VariableGenerator ó ResortSim	Pseudo-random variables are generated and fed into the ResortSim function for all instances where an element of chaos is to be accounted for.	Chaos Maker
ResortSim ó GraphDisplay	ResortSim generates and returns an appropriately selected graph to the GraphDisplay based upon the user’s input for GraphMode.	Graphical Output
ResortSim ó GUI	ResortSim returns any numerical values to the GUI to be displayed to the user after simulation has taken place.	Output Generation

Attributes

Concept	Attributes	Attribute Description
All Input Variables	GameMode	Determines which set of input variables to consider for simulation.
	NumIndividual	The number of agents participating in the game.
	IterNum	The number of rounds played before the end of the game.
	Temp	Roughly determines whether actors will be drawn to the mountain.
	FriendsAvg	The average number of people grouped together whose opinions will influence each other's.
	GamePerc	The parameter that will be used as a benchmark to determine whether or not a given iteration was successful.
	DecPerc	Determining what percentage of an agent's decision to attend is strategy.
	ExpPerc	What percentages of actors are skillful, and therefore will be more likely to attend.
	LowerBound	The lower limit in deciding the probability of whether or not.
GameMode	Individual	The simulation is run including use cases 1 – 6; factoring in such variables as NumIndividuals, IterNum, Temp, FriendsAvg, GamePerc, and DecPerc.
	Ski Resort	The simulation is run including use cases 7 – 13; factoring in such variables as NumIndividuals, IterNum, Temp, FriendsAvg, GamePerc, ExpPerc, and LowerBound.
GraphDisplay	GraphMode	Determines which data is to be graphically output by the GUI after the simulation.
	GameMode	Determines which set of input variables to consider for simulation.

Traceability Matrix

Domain Concepts

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
UC-1	X															
UC-2		X	X	X	X	X		X	X							
UC-3		X	X	X	X	X	X									
UC-4										X	X	X	X			X
UC-5														X	X	
UC-6														X	X	
UC-7												X				X

System Operation Contracts

UC-1:

Operation:	Startup
Preconditions	None
Postconditions	Start screen is displayed with choices for either Individual or Ski Resort.

UC-2:

Operation:	SkiResortFields
Preconditions	Ski Resort mode of operation has been selected.
Postconditions	• `gamepercent != NULL` • `numagents != NULL` • `simlength != NULL` • `weather != NULL` • `avgnumfriends != NULL` • `exppercent != NULL` • `lowerbound != NULL`

UC-3:

Operation:	Individual Fields
Preconditions	Individual mode of operation has been selected.
Postconditions	• `gamepercent != NULL` • `numagents != NULL` • `simlength != NULL` • `weather != NULL` • `avgnumfriends != NULL` • `decpercent != NULL`

UC-4:

Operation:	RunSimulation
Preconditions	All data entries have been completed.
Postconditions	• `overallsuccess != NULL` • `msuccessful != NULL` • `lsuccessful != NULL` • `magent != NULL` • `lagent != NULL` • Graphs are displayed and available for viewing.

UC-5:

Operation:	Individual Graphs
Preconditions	Simulation runs successfully and all values have been properly calculated.
Postconditions	Graphs are displayed and available for viewing.

UC-6:

Operation:	SkiResortGraphs

Preconditions	Simulation runs successfully and all values have been properly calculated.
Postconditions	Graphs are displayed and available for viewing.

UC-7:

Operation:	SaveFile
Preconditions	Simulation runs successfully and all values have been properly calculated for at least one individual run.
Postconditions	Current values in the present state are saved to a file.

Mathematical Model

The foundation for this project is the El Farol Bar minority game. The classical example features bar patrons who will not have a successful night if the bar is overcrowded such that more than 60% of people attend, or if they didn't attend when the bar was not crowded. More interestingly, actors may maintain a memory of previous experiences to determine whether or not they should attend the bar on a given night. Likewise, the actors surrounding each other will have influence on whether or not a good time is being had.

Our model for the ski resort simulator takes many basic elements from the classical minority game and expands upon them. The resort won't stay open for business for very long if the number of people that attend is too high, because patrons will experience overcrowding on the slopes and may decide not to come back. This parameter will be set in the beginning of the project by the user, and will be referred to as the Game Percentage. Other parameters set by the user will be number of individuals/skiers, length of the simulation weather, average group size (average number of friends), and skier's skill level. Since our program supports 2 types of users, Ski Resorts and Individual Agents, they each have extra parameters that only they can set. For example, only Ski Resorts can set the Experience for the first day, to see how much a bad opening may hurt their outlook, and only individual skiers can set the Decision Percentage, which determines what percentage of their decision is based on strategy. The first parameters will be quantified and combined to determine how appealing the ski resort is on a given day and whether a skier will go. Yet another parameter, IQ, will be discretely uniformly distributed from 1 to 4 in order to have more data points at the highest and lowest IQ levels so that the graphs we produce at the end will be more reliable. This coupled with the assigned Decision Percentage will allow us to compare success rates with IQ more directly. Higher IQ individuals will have a longer short-term memory and will choose to base a greater percentage of their decision on strategy than others in order to increase their success rate.

Values such as IQ will be static from the beginning of the simulation, but values such as weather and experience will vary every week based upon their underlying normal distributions. Skill level will also increase slowly over time.

As the user inputs these parameters, they will only be accepted as valid if they are formatted in this way:

- Game Percentage must be between 10 and 90
- Number of Agents/Skiers must be at least 100
- Length of Simulation must be at least 10 weeks (so that strategies begin to fall into place and become more accurate)
- Weather/Temperature must be between 0 and 60 (Fahrenheit), and each week's weather will be a normal distribution with a mean as the set weather, and a standard deviation of 10 degrees whereas each skier's weather preference will be based on a normal distribution with mean 30 and standard deviation 10 (if the week's weather is better than the individual's weather preference, then they are more likely to go)
- Average Number of Friends must be between 1 and 8, and each skier's number of friends will be decided at the beginning as a normal distribution of mean the set average and standard deviation 2
- Experience Percentage is set by the ski resort and should be between 0 and 100

- Skill level is set by the ski resort and is on a scale of 1 to 10 (1 being novice and 10 being expert)

- Decision Percentage is set by an Individual and should be between 20 and 80 percent, to allow for the shifting that is done by each skier based on their IQ

A variable "decision" will be quantified for each actor. If the value of decision is greater than or equal to 0.5 (since it will be between 0 and 1), then the actor will decide to attend. The decision is quantified in this way:

Decision = Decision Percentage * (Strategy) + (1 – Decision Percentage) * 0.25 * (friends + weather + skill + past experience)

Where:

- Decision Percentage is a real number between 0 and 1 (set by the user in the beginning if the user is an individual, and set by the system if the user is a Ski Resort), this value is slightly adjusted based on the individual skier's IQ (for example, if it was set at 50%, a skier of IQ 4 would take it as 70%, whereas a skier of IQ 2 will take it as 40%)

- Strategy is either a 0 or a 1 (a yes or a no), determined by the underlying strategy used by this individual, which depends on the Game Percentage that was set and the individual's IQ

- (1 – Decision Percentage) is to make the decision weighted so that the higher the decision percentage, the less important external factors are, and vice versa

- Weather will be based on the person's weather preference and that particular week's weather such that if their preference is for weather warmer than that week's weather, this value will be 0, whereas if it will be warmer than their preference, then this value will be a 1

- Skill will simply be that skier's skill level divided by 10 to ensure it is between 0 and 1

- Experience will be between 0 and 1 and based on their own personal experience during the last week they went

- The friend field (F) in the equation above will be between 0 and 1, representing a total of the experiences of those friends who attended during the last week. This influence can be expressed in this way:

  $F = \sum_1^n x(j) * f(j)$

  Where:
  n = the number of friends the individual has.
  f(i) = whether friend i went skiing last week (0 or 1).
  x(i) = friend i's experience last week (real between 0 and 1)

  This setup allows for a situation in which a skier with more friends who had a good time will be more 'persuaded' to attend this week as is the case in reality.

Strategy Outline

The strategies agents have depends on their IQ. IQ is uniformly distributed so that one-fourth of all individuals have each IQ level (1 to 4).

The basis of a strategy whether to go skiing or not depends on a combination of 2 things, the raw logical strategy determined by analyzing the ski resort's success for the past few days, and an individual's personal feeling about whether they should go based on their own recent attendance. Since the latter is less accurate, the higher the IQ of a skier, the less they take this into account. The reason we considered this factor is because for example, if one's logic says that they should go skiing on a particular week, the fact that they went the past 3 weeks in a row may make them change their mind. Also, since having a longer short-term memory produces better predictions based on the ski resort's recent success rate, the higher one's IQ, the longer their short-term memory is. This breakdown of how much of the strategy is logic and how much is personal motivation is broken down in this way:

IQ	Part A) Self / Personal Motivations based on own recent attendance	Part B) Pure Logical calculation based on ski resort's recent attendance percentages
1	75%	25%
2	50%	50%
3	25%	75%
4	0%	100%

Part A of the strategy process is based on a chart like below (an individual of IQ 3 looks back 3 days, so 010 for example means that this individual did not go 3 weeks ago, went 2 weeks ago and did not go last week).

Previous Weeks	Part A Conclusion
000	1
001	0
010	1
100	1
011	0
101	0
110	1
111	0

In the same way, an individual of IQ 1 only looks back one day for both Part A and Part B of the strategy, thus resulting in a less intelligent decision than those of higher IQs.

Part B strategy is determined by the decisions of lower IQ's (since an IQ 2 skier will essentially be smarter and will be able to calculate IQ 1's strategy and act on it) as well as the ski resort's recent success rates. Once again, those of higher IQ's have greater knowledge about the general population on top of a longer short-term memory. The logical strategies of an IQ 2 individual, for example, include the following:

- If IQ 1 people would decide not to go, then I will go.
- If IQ 1 people would decide to go, then I will go only if the attendance percentage of 2 weeks ago was above the attendance of last week, but below the set game percentage.
- Otherwise do not go

This logical strategy combined with the personal-motivation conclusion, as broken down by the percentages given on the previous page, decide whether an individual will go or not. Undoubtedly, the higher the IQ, the more complex the conditions above become. One strategy, of many, for an IQ 4 individual may be:

- If 2 of the other 3 lower IQ groups would decide to go, then I will go only if the game percentage is greater than 0.5 (since 25% of the population is made up by each IQ level).

We believe that this way of balancing personal motivations (not appearing selfish by not wanting to go 3 weeks in a row or feeling entitled to go if one has not gone for 3 weeks), and intelligence (longer short-term memory, more complex conditions on whether to attend, the ability to determine what lower IQ skiers would do, and the decision to make logic a bigger part of one's strategy) is a fairly accurate and realistic model for strategy.

Plan of Work

After submitting Report 1 on Friday, February 17, 2012, we will meet to assign responsibilities and begin software development and testing. We will split up the various sections, including Class Diagrams and Interface Specification, System Architecture and System Design, Algorithms and Data Structures, and User Interface Design and Implementation.

Tentative Dates:

February 18 – Start work on e-archive, assign responsibilities

Individually work on assigned sections, while collaborating when needed.

March 3 – Begin compiling individual sections, and preparing Report 2

References

1. Software Engineering by Ivan Marsic
2. El Farol Bar Problem - http://en.wikipedia.org/wiki/El_Farol_Bar_problem
3. El Farol Bar Problem and the Minority game Project Description - http://www.ece.rutgers.edu/~marsic/books/SE/projects/MinorityGame/
4. National Skiing Statistics - http://www.nsaa.org/nsaa/press/facts-ski-snbd-safety.asp