**Design Specification**

**Design Consideration**

*assumptions and dependencies*

 The user will be comfortable using a keyboard, running a desktop computer. Our file will run on python and execute its function within a minute. We will finish our code by 6/15/15.

*goals and guidelines*

We aim to have our code finished by 6/15/15. We also aim for our code to run with limited space and be user friendly.

**Architectural Strategies**

The input will all be excel files because this is more user friendly, if not at least more often used when storing data than other types of files. The outputs for our code will include a table and the option for a couple types of graphs. The table will provide the user with the actual numbers resulting from the data analysis. The option to provide the several different graphs will be to accommodate the needs of our user and their data presentation preference. In detail, the system will analyze an ecosystem where the user inputs four parameters (production ratio, consumption ratio, addition, and removal) of a set list of species. From these parameters, individual biomasses will be calculated by solving the system of linear equations. The output (the biomasses) will be given in a table format with an option to also see it in graph(s).

**System Architecture**

We will have four systems: UserInterface, ParameterRepository, EcosystemAnalyzer, and EcosystemAnalyzerOutput. In chronological order of use, we first have the UserInterface. This will prompt the user to input an Excel file. The ParameterRepository will convert the Excel file input to be read by python. Next, the EcosystemAnalyzer will solve linear equations as data analysis. Finally, the EcosystemAnalyzerOutput will format the output of EcosystemAnalyzer into a table and graph format for the use to access.

**Policies and Tactics**

We will follow the Google’s Python coding style guide and our code will be available for our CSE490 classmates through GitHub.

**Detailed System Design (Interactions flow chart available at the very bottom of section)**

*UserInterface*

**Classification:** Function

**Definition:** Code will ask the user to input an Excel file of 4 different parameters ((P/B)I or production ratio, (Q/B)i or consumption ratio, removal, and addition) and identify how the output should be displayed (graph type).

**Responsibilities:** Prompt the user to input an Excel file using the terminal given short directions. The user will be given an option to download an Excel template that already have the headers. If a detailed direction is wanted, then the user will be able to call for this.

**Constraints**: The user must input an Excel file with the right format.

**Composition**: This section will have many if-loops that allows the user to call for more directions or to access an Excel template, if desired. This will also allow for a step-by-step direction where with each step, new directions can be provided.

**Uses/Interactions**: The Excel file that the user inputs will be called for and used in ParameterRepository. UserInterface itself is called when the user accesses the whole file through the terminal. In other words, the user will not have to specifically call for UserInterface in any way as it will be automatic. See above interaction web diagram.

**Resources:** User input of Excel file(s).

**Processing**: Will contain several if-loops to provide help/detailed directions, make Excel template available and to check if input follows wanted format.

**Interfaces/exports**: This section will have an Excel file stored (for the user to download) and then allow for the user to input another Excel file (with a specific format, determined by the order of the headers and whether there are floats in all of the required cells).

**Detailed Subsystem Design**: UserInterface is a function that provide directions and a template Excel file for the user to use to input a filled, and correctly formatted, Excel file. The UserInterface will have a series of if-loops to provide the directions, provide file template, and check if input is correctly formatted. The user must be familiar with the terminal and Excel to utilize this system.

*ParameterRepository*

**Classification**: Function

**Definition:** Access the Excel file with the 4 parameters ((P/B)I or production ratio, (Q/B)i or consumption ratio, removal, and addition) and transform the input into python-friendly format. There will also be a string specifying how the user desires the output should be graphed (format determined in template Excel file).

**Responsibilities:** Transform Excel file user inputs and make it available for data analysis in EcosystemAnalyzer. This will involve

**Constraints**: User must have xlrd and numpy installed.

**Composition**: This section will use xlrd to read the Excel file. The data will be read as floats and strings. Several scalars will be made to store values corresponding to the different parameter types that have been input through the Excel file ((P/B)i, (Q/B)j, Ri, and Ai) as well as a list for output specification.

**Uses/Interactions**: The Excel file in UserInterface will be accessed by ParameterRepository. This data, formatted into a float, will be accessed by EcoSystemAnalyzer for data analysis.

**Resources:** Access to xlrd and numpy.

**Processing**: Will contain xlrd and while loops (to go through the read lines to format into a float). Numpy will also be used to make scalars containing different parameter types ((P/B)i, (Q/B)j, Ri, and Ai). A list for output specification will also be made. Additionally, since the format of the Excel file input is specified, the headers will not be read from the file. Instead, a scalar specified (ie. named) for each parameter will be made/filled.

**Interfaces/exports**: This section will input an Excel file, through UserInterface and then output a string, for graph specification, and 4 scalars: (P/B)i, (Q/B)j, Ri, and Ai.

**Detailed Subsystem Design**: The ParameterRepository is a function that reads the Excel file input (from UserInterface) and transforms it into a python-friendly format of strings and floats. The parameters necessary for data analysis will be made into multiple scalar matrices, grouped by parameter type while the string will hold information for how the user desires the graphical output. The scalar matrices of the parameters will not be sorted as the order will correspond to the set order of species, which is conserved between all matrices. For ParameterRepository to function properly, it must be able to access xlrd and numpy. Moreover, this section is contain functions from xlrd and numpy integrated in while loops.

*EcosystemAnalyzer*

**Classification**: Class

**Definition:** Using the parameters input from the user to solve the linear equation for biomass.

**Responsibilities:** Code will access the 4 parameter scalars ((P/B)i, (Q/B)j, Ri, and Ai) to solve the linear equation for biomass or Bi formatted in a class.

**Constraints**: User must have numpy installed.

**Composition**: The EcosystemAnalyzer will format the 4 parameters scalars and a placeholder for Bi into a series of matrix to solve using systems of linear equation.

**Uses/Interactions**: The 4 scalars from ParameterRepository will be accessed. The solved Bi will be later accessed by EcosystemAnalyzerOutput.

**Resources:** Access to numpy.

**Processing**: Will join various scalars together to form matrices. A placeholder for Bi will also be made (an empty scalar). Furthermore, systems of linear equation will be solved using numpy.linalg.solve().

**Interfaces/exports**: This section will input 4 scalars, through ParameterReposotiry, and then output Bi.

**Detailed Subsystem Design**: EcosystemAnalyzer is a class where all the parameters will be held and analyzed. The EcosystemAnalyzer will access the necessary input from ParameterRepository and the output will be a single scalar calculated by solving the system of linear equations. The user must have numpy installed on the computer and accessible by python, as solving system of linear equations efficiently requires it.

*EcosystemAnalyzerOutput*

**Classification**: SuperClass

**Definition:** Given Bi, or biomass, provide the user with a table and make visual representations (graphs) of this of it available.

**Responsibilities:** Make a Bi table and line and bar graphs for user to access.

**Constraints**: User must have numpy installed.

**Composition**: The EcosystemAnalyzerOutput will make and display a table with Bi. There will be a while and an if-loop to go through the strings from the ParameterRepository to indicate which graph will displayed.

**Uses/Interactions**: The scalar of Bi from EcosystemAnalyzerOutput will be accessed. The string indicating graphing preference from ParameterRepository will also be accessed.

**Resources:** Access to numpy.

**Processing**: While and if-loops will be implemented to identify how the data will be graphed. The Bi scalar will be accessed to be made into table.

**Interfaces/exports**: This section will input a scalar (Bi) and a string, through EcosystemAnalyzer Output and ParameterReposotiry, respectively. The output will be a table of Bis and possibly graphs of Bis, if indicated by the user.

**Detailed Subsystem Design:** The EcosystemAnalyzerOutput will be a SuperClass of the EcosystemAnalyzer class. This SuperClass will generate a table of resulted biomasses and visual representations (graphs) if prompted by the user. In order to use this SuperClass, the user must have numpy installed. There will be a while and an if-loop to go through the strings from the ParameterRepository to indicate which graph will be displayed. The scalar of Bi from the EcosystemAnalyzerOutput will be accessed. The string indicating graphic preference from the ParameterRepository will also be accessed. This section will input a scalar (Bi) from EcosystemAnalyzerOutput and a string from the ParameterRepository. The output will be a table of Bis and possibly graphs of Bis, if prompted by the user.

**Note:** After the user inputs one Excel File and acquires the output (calculated biomass), the system will loop back to UserInterface. This loop occurs to allow input of a maximum of 3 Excel files. UserInterface will display the same information as before, however the template Excel File will be what was previously submitted. The following cascade of system interactions will be the same as previously, however the graphs will incorporate the data for all the previous calculated biomasses.

