Architecture and Rationale

Two Major types considered: Pipe and Filter and Event Driven

Type Chosen: Event Driven
Reason for the chosen type: For this project, the key ingredient is not the program's looks, speed, or mobility. The key aspect is the successful conversion from an image into a spreadsheet-readable format such as CSV. Because of this there was a moderate debate between a Pipe and Filter based design and an Event Driven based design.

An event driven based design is usually stronger when the user is very active with the program, and the program itself is what the user is interested in. For our program, the user is only interested in getting the data converted as fast as possible and then saving that data out to a file. Nothing runs in the background if the user does not click on a button, so it makes sense to implement an architecture that waits for user input.

A Model-View-Control environment allows for us to focus much more on the data end of the program without worrying about what GUI will be implemented and also allows for things such as command line control (if later requested by the customer).

The key components of the program include:

- Image -> CSV
- CSV -> Image
- Large Image -> Smaller Image
- JPG-> BMP (any type conversion)
- CSV-> 3 spreadsheets

Why not a pipeline: We chose not to use a pipeline method because we simply did not have enough filters to justify this sort of architecture. We only have about 3 filters and 2 different types of data (image and array). Pipe and filter would have been more convenient for later implementing a command line version, but the customer did not want this functionality. Also, since we have a relatively small set of core functions, the translation to a new language will not be difficult.

Data

Data Storage While the Program Runs

The images are stored in 1 of 2 formats in the computer's memory:

- QImage : a native format of Nokia's QT platform that can easily be displayed in one of their GUI's

The spreadsheet is stored as:

- A vector <xyrgb> where xyrgb is a struct of 5 char values
- The vector format allows a CSV file to be read in easily without first determining the valid line count
- The max of the x and y positions are used to initialize the proper image size

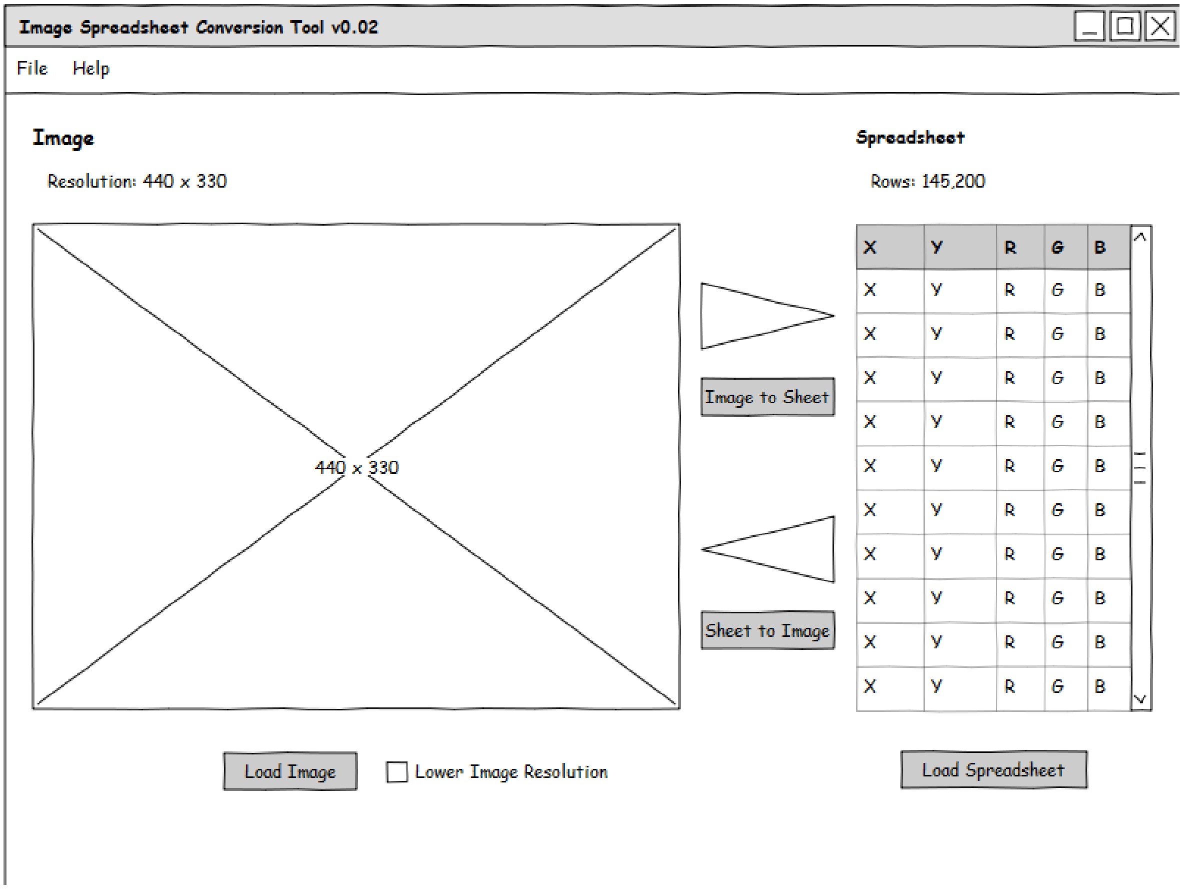
Data Storage Outside of the Program

Data will be stored in two sets, images and text files. Images are a standard type of data with set conventions that we will use (BMP,JPEG, PNG, etc). The text files will be in a CSV (comma separated value) file initially. This format is compatible with both Excel and OpenOffice and is simple to write to.

CSV was chosen for several reasons over .xls:

- Simple to read/write to
- Easily Imported into Excel or OpenOffice
- Readable in a text editor

GUI



X	Y	R	G	B
X	Y	R	G	B
X	Y	R	G	B
X	Y	R	G	B
X	Y	R	G	B
X	Y	R	G	B
X	Y	R	G	B
X	Y	R	G	B
X	Y	R	G	B
X	Y	R	G	B
X	Y	R	G	B

Validation

We plan to meet with our customer once per week to check our progress against his expectations. We began with GUI mockups to receive feedback on the interface of the program. Later, we demonstrated a working prototype for feedback on functions. Afterwards we After he has tested the program, we will be able to cyclically better the program.

Also, as our end-users will be students, we hope to visit a high school teaching conference and view students testing out our software in person to make sure that they understand its function and how to use it.