**UWROV GUI Documentation**

**Contents:**

[to be completed]

**Preface:**

Our goal for this new iteration of the graphic user interface is to have a reusable, updatable, manageable user interface that can be used for future years. This will ultimately imitate a company setting where new hires will usually “update” the current software rather than remaking new versions of the software every time. This will require persistent documentation and a well-trained team. I hope that everyone will work together to provide a productive learning environment even many years in the future.

I am sure that not many of you have experience working on team programming projects especially since there are not many opportunities outside of a company environment to work on massive projects with multiple people. This approach to projects should help develop skills to work with people of many different backgrounds of programming and skill level.

To make sure that we have a manageable and flexible User interface, we will adapt a specific project architecture that will make sure that our project remains organized and manageable.

Our design philosophy will focus on the idea of “modular programming”. This style of programming focuses on separating functionality of a program to individual, independent “modules” that can be modified separately, and switched out without affecting the rest of the program as a whole. When a program is small, it is not very difficult to tell everyone to learn how the program functions as a whole. However, the program develops and gets to a point where it becomes unreasonable to tell everyone working on it to go and learn how the whole program functions. By adapting a modular structure, programmers will only need to focus on each individual “module” that requires only the knowledge of how the module interacts with the “core” and how the module acts. A good modular architecture will lessen the workload of managing clunky and poorly structured programs.

A crude analogy of how a well-modulated program will function is the human body. If an organ in a human body loses function or malfunctions, the only work required to fix it is to fix or replace the organ that is having problems. It is unnecessary to touch the other organs that are working fine. All the organs have their own inner workings and have their individual functions. It would be very difficult to remedy an organ if it had more than one core function. For example, let’s say that an organ had the functionality of a heart and the stomach but the stomach functionality somehow malfunctions. It is cumbersome to try to fix the stomach without messing up the heart functionality of the organ. In conclusion, you would need to work around the heart functionality and hopefully not mess up that functionality to fix the stomach functionality. All in all, it is very efficient to manage separated modules rather than have one that has interconnection throughout all the functions.

**Architecture:**