**Team – B – SE 362 Unit 1**

**Design Document and Rationale**

**The initial Design of our Drawing Program is modeled to accept extension easily. Our most important focus is to provide a base on which any type of Glyph or operation on a Glyph could be programmed to the interface we provide. In this initial design the implementation of classes is not included, and has not been discussed beyond the interaction of the different objects at runtime.**

**Our design implements 3 different GoF patterns: Composition, Proxy, and Strategy. We chose to use Composition for the Glyph and Glyph Group to simulate a recursive hierarchy of “Glyphs” to be drawn on the screen. We chose the Command pattern for the Document to Operation relationship to separate the implementation of certain functions that can be performed on Glyphs from the Glyphs themselves. By using Command we have given ourselves a simple way to extend the functionality of operations in the future. We chose the Proxy pattern to stand in the place of the Document for the GUI and the FileParser because we wanted to route the requests from the GUI and FileParser through the proxy to hide the implementation behind the Document.**

**Using Command as the pattern for the different operations on Glyphs was a choice to, again, promote the easy extension of functionality in the future. The concept behind the decision is that given the 2 important points on any Glyph the operations can manipulate those points, and the Glyphs themselves will be able to redraw themselves based on the new points. The Command pattern allows us to hide the implementation of any operation behind a common interface, thus allowing for the simple construction of a new operation and simple integration of the new operation.**

**The Composition pattern allows us to have the groupings required by multiple Glyphs. The concept behind this decision is that we needed to have groups and sub groups of Glyphs and the composition pattern allows for that functionality easily.**

**Proxy Pattern allowed for some implementation encapsulation, and also for the separation of request routing and the answer to the requests themselves. The Proxy will route certain requests to the Document letting the document handle the requests without worrying about what the request is. This design is more cohesive because document will have specific responsibilities and not be handling things that are outside of the skill set we should expect from it. The implementation encapsulation seems a nice bonus for the choosing of this pattern and should allow for simple extension or change of functionality in document without the GUI and FileParser needing to adhere to the new changes.**