ChronoTimer Project

Group – Zero Cool

**Iteration Plan**

Sprint 0 – Our group’s plan for Sprint 0 is to ensure that we have everything logistically in place for us to get started on Sprint 1. This entails configuring a gitHub repository for source control, splitting up the expected work load into individual tasks and insuring that there are effective lines of communication set up between the group.

During Sprint 0 we also want to set a time weekly where we can meet briefly outside of just email or phone communication which is laid out in the group charter. We also want to identify what timing mechanism we are going to be using for the ChronoTimer.

Sprint 1 – The group’s plan for this iteration is to have our Chrono Timer project meet the requirements laid out in the project document and have those software requirements fully tested before this iteration’s release according to our test plan. The main requirement of the system for this iteration is to have one stream of individual racers whom can start, finish, cancel or not finish on an individual timed basis. GUI is not required for this iteration’s release.

Sprint 2 – Our plan for Sprint 2 is to meet the requirments made in the project document and make sure they are tested according to our test plan. Requirements for this iteration include exporting of event data to a file on a USB stick and to also have that data displayed on the console. We also need to add the ability for multiple channels to be able to be used this time around instead of only 2 from Sprint 1 release.

The ChronoTimer will also be getting some GUI incorporated with it this iteration so we want to focus on this new aspect and really make sure it is tested properly. We also want to attempt to look back at Sprint 1 and see what went wrong and how we can improve for this Sprint. We also need to implement the other event types besides individual, so we plan on using the same method we derived Individual from a parent “Abstract Event” class.

Sprint 3 – The plan for this final release is to handle single start with multiple finishes, additional maintenance including advanced GUI and multiple displays and to incorporate some more advanced data analytics.