**Project Evaluation**

For a Track & Field Meet Server

Version 1.0

Submitted in partial fulfillment of the requirements of the degree of MSE

Tracy Marshall

CIS 895 – MSE Project

Kansas State University

# Table of Contents

[Table of Contents 2](#_Toc469174212)

[1. Introduction 3](#_Toc469174213)

[2. Problems Encountered 3](#_Toc469174214)

[2.1. Learning Existing Techniques 3](#_Toc469174215)

[2.2. Determining Project Requirements 3](#_Toc469174216)

[2.3. Time Management 3](#_Toc469174217)

[3. Metrics 3](#_Toc469174218)

[3.1. Source Lines of Code 4](#_Toc469174219)

[3.2. Project Duration 4](#_Toc469174220)

[4. Lessons Learned 8](#_Toc469174221)

[4.1. Cryptography and Security 8](#_Toc469174222)

[4.2. Server Implementation 8](#_Toc469174223)

[4.3. Programming Language 8](#_Toc469174224)

[5. Future Work 8](#_Toc469174225)

[5.1. Build up framework 8](#_Toc469174226)

[5.2. Work On Security 8](#_Toc469174227)

[5.3. Dry run 8](#_Toc469174228)

# Introduction

The purpose of this document is to perform a post analysis of the Track & Field Meet Server (TFMS) development. Things that will be covered will include the problems that were encountered during development, lessons learned, and the future for the TFMS project. Project performance metrics will also be aggregated and summarized at the end of this document.

# Problems Encountered

The following section outlines and describes many of the big issues that were faced with having a successful project.

## Learning Existing Techniques

The number one problem I faced was determining the best way forward with many pieces of my design. I chose a language that I was not the most familiar with in an attempt to grow my knowledge about another programing language. I was able to learn a lot about python but this ended up slowing me down in many ways. I had not dealt with servers in my previous work so trying to determine what pieces I could pull from libraries that already exists was very difficult. I was unfamiliar with most of the current techniques so a lot research had to be put in to determining what would provide the least amount of friction and the most amount of flexibility for what I needed. I overwhelmed myself early on with trying to produce a server with perfect crypto and this lead me down a lot of dead ends. I also spent a lot of time examining data storage strategies before I settled on using a SQL database. This was the first time had I had utilized SQL so there was a bit of a learning curve with that as well.

## Determining Project Requirements

The second biggest problem for me was requirement creep. I had trouble focusing on what my core functionality was. I kept looking at the TFMS at a larger scope of implementation and it quickly began to have so much going on that I would distract myself from what was really important. As I got more into the functionality I found that it would have been better if I scoped my project to a subsystem of the TFMS and spent more time building a framework for that subsystem to support a theoretical future TFMS.

## Time Management

The final problem I faced was time management. The first phase of the project wasn’t too bad as I took it over the summer with no other classes to distract me. Then my wife and I had a baby at the end of the summer and my distractions increased. Throw that on top of doing the second and third phase at the same time as a security class and working full time during an implementation heavy phase for my company and it time quickly disappeared. I unfortunately had to prioritize the security class over the project on many of my weekends where I had many available hours to work because of deadlines in that class. I worked through the prioritization process and was able to get through the TFMS project but was left with far less hours than I would have liked to have had to spend on the project.

# Metrics

This section looks at metrics that were collected during the execution of this project.

## Source Lines of Code

|  |  |
| --- | --- |
| **Phase** | **SLOC Estimate** |
| Phase 1 | 100 |
| Phase 2 | 275 |
| Phase 3 | 325 |

Table 1- SLOC by phase

The table above shows the amount of SLOC that was accomplished by phase. These numbers fell short of my original estimate of 1KSLOC but given the road blocks that were overcome were still acceptable.

## Project Duration

|  |  |  |
| --- | --- | --- |
| **Phase** | **Start Date** | **End Date** |
| Phase 1 | 6/6/16 | 7/29/16 |
| Phase 2 | 8/22/16 | 10/24/16 |
| Phase 3 | 10/24/16 | 12/12/16 |

Table 2- Project Start and End Dates

Table 2 shows the start and end dates of the different phases of the TFMS project. The first phase was conducted during the Summer 2016 semester and the gap from Phase 1 to Phase 2 indicates the break between semesters. The final two semesters were completed during the Fall 2016 semester. The figures below show my progress through each phase.

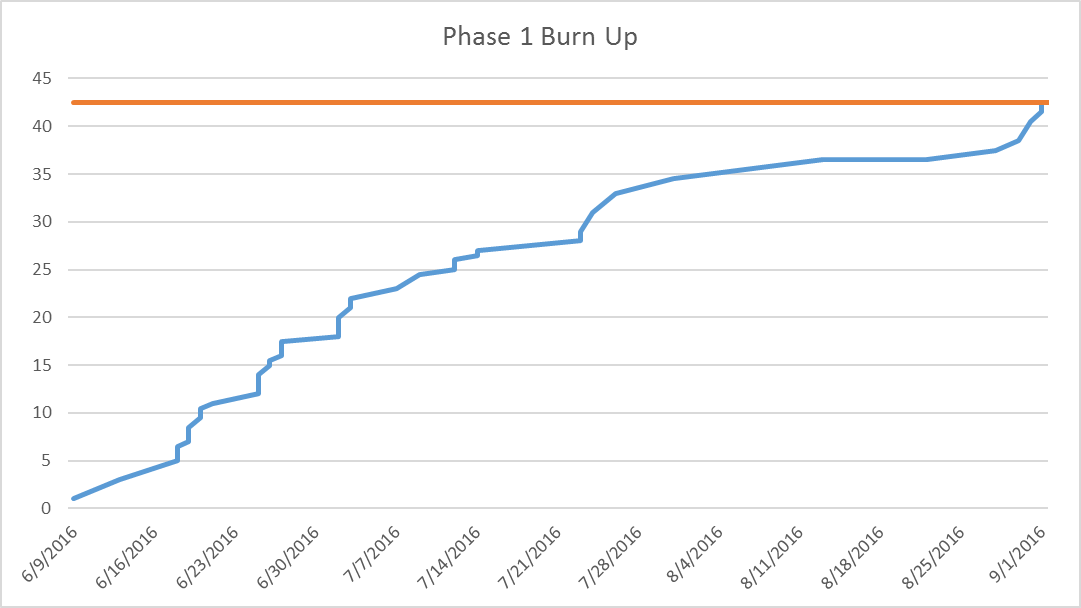


Figure 1- Phase 1 Burn Up

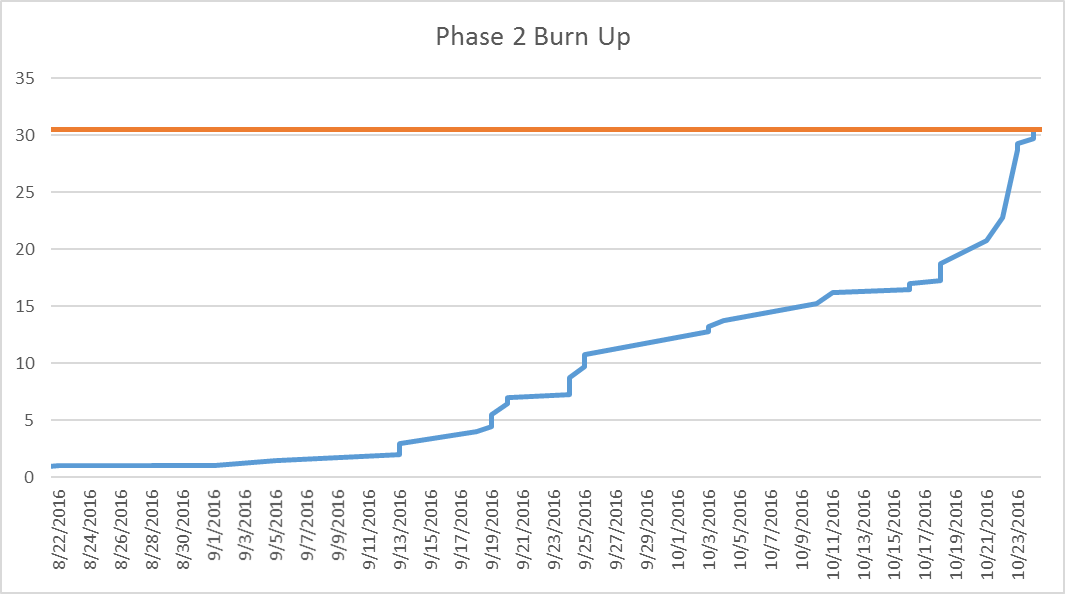


Figure 2- Phase 2 Burn Up

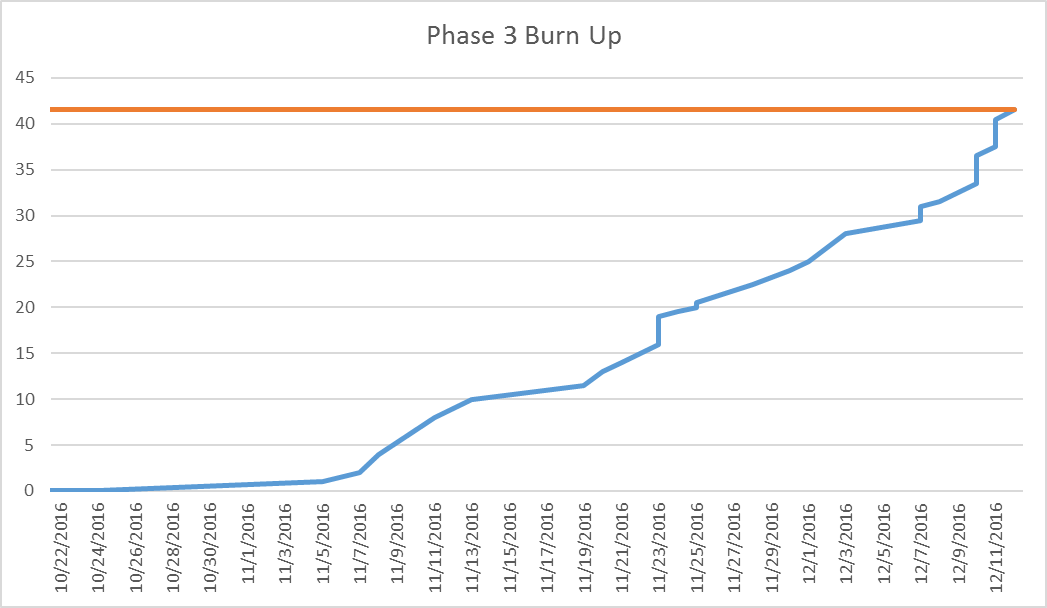


Figure 3- Phase 3 Burn Up

The graphs below show what the distribution of work was during the entire project. First I’ll show what the break out is by phase and then I’ll dive into each phase to see how the work was distributed across the different portfolios of work.

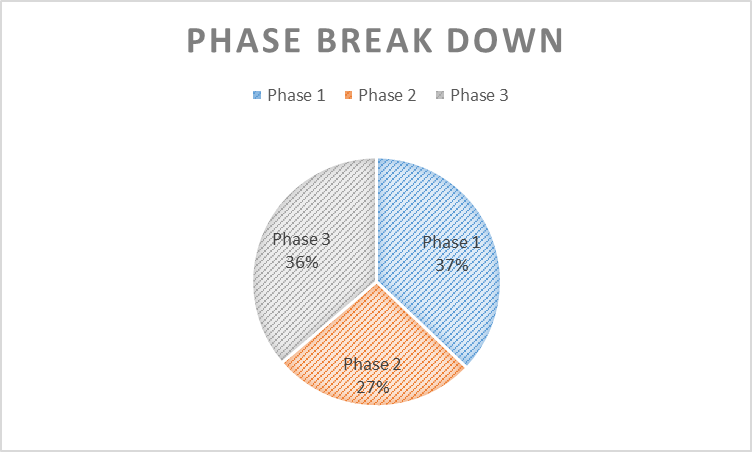


Figure 4- Break down by Phase

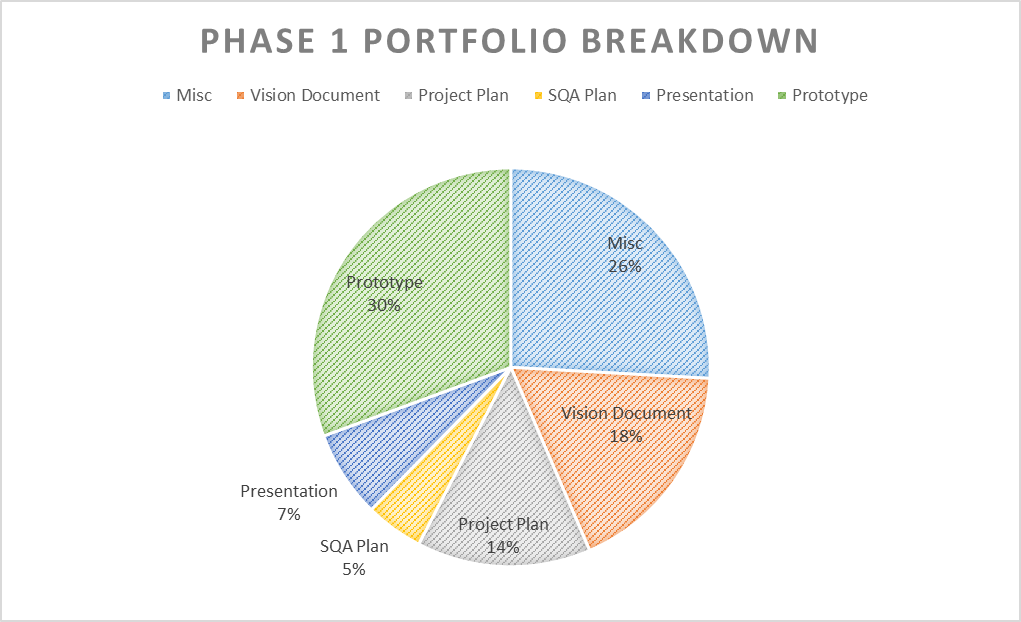


Figure 5- Phase 1 Portfolio Breakdown

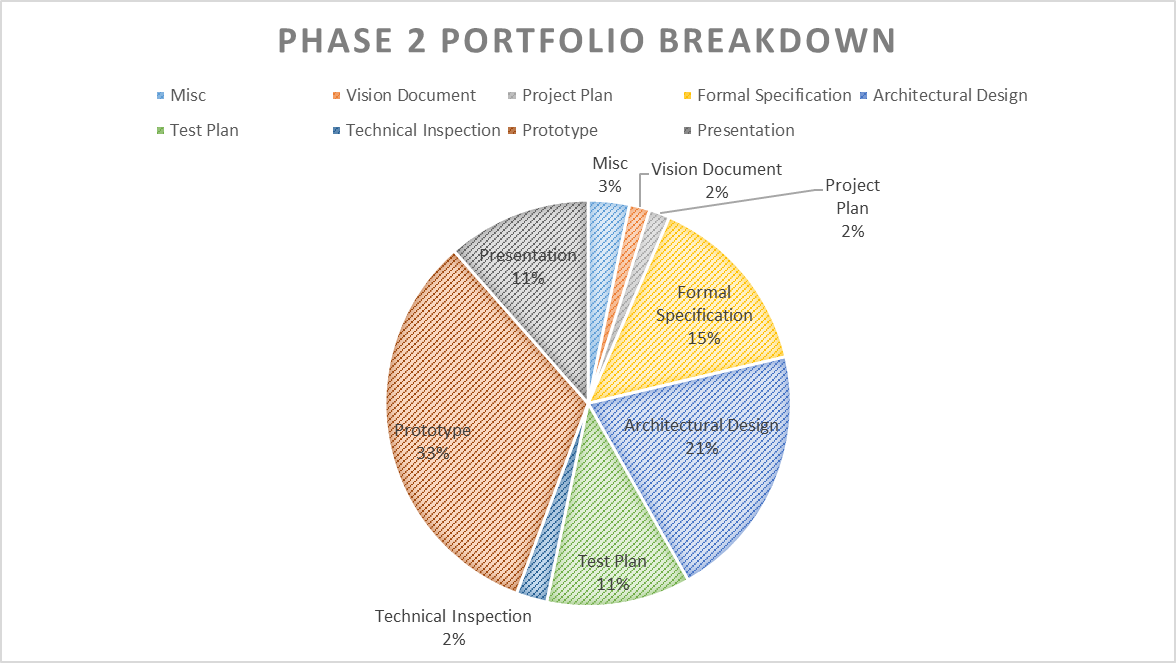


Figure 6- Phase 2 Portfolio Breakdown

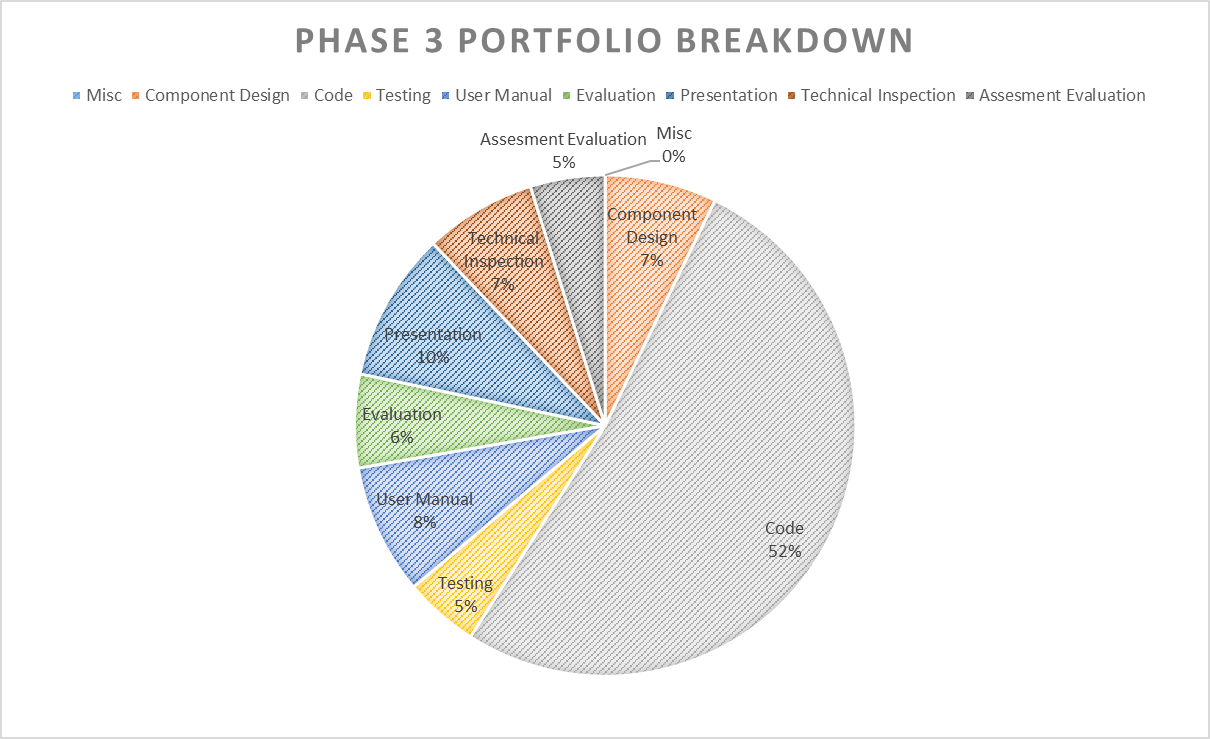


Figure 7- Phase 3 Portfolio Breakdown

The final figure takes a look at how well I was able to estimate my work. These metrics look at what sort of effort I initially planned for the given phases and then compares that to how much effort it actually took me. An efficiency score of over 100% indicates that I was more efficient than expected, meaning it didn’t take me as long as initially planned.

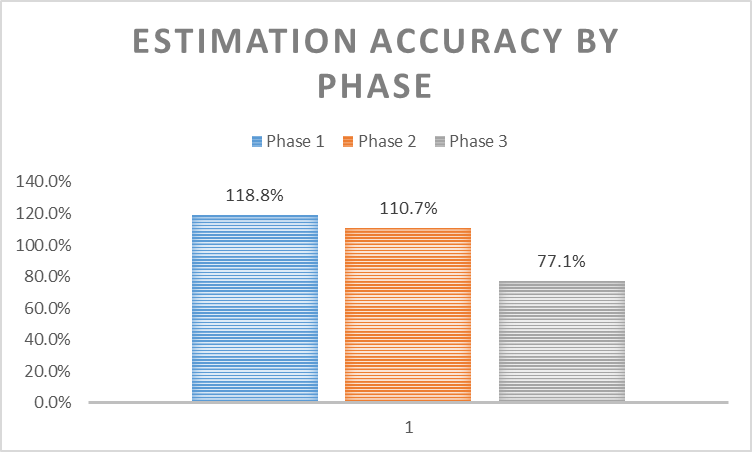


Figure 8- Estimation Accuracy by Phase

# Lessons Learned

## Cryptography and Security

A big lesson learned was taking for granted good security. I believe that to have a secure system that it can’t be an afterthought, but it is certainly difficult to do when you have not practiced many of the security concepts. I would definitely say that for a prototype for a project that is still trying to mature its initial design that it is much simpler to consider security after the foundation has been laid. I was often distracted by trying to perfect the security of my system that I wasn’t able to make as much progress as I would have like. I also learned that there are a lot of possible solutions and not all of them are as easy as others, so understanding the possibilities sooner would have been better.

## Server Implementation

Another lesson learned was that a lot of work as already been put into developing server frameworks. I haven’t used any of these and had little knowledge of them so I began down a path of designing much of the low level implementations of the server. This became quickly overwhelming with the amount of effort it was going to require. It wasn’t until late in the game that I found some classes that I could use to build my server and if I would have more time I could have found a more comprehensive solution to avoid lost time on development and more time on integrating.

## Programming Language

My final lesson learned was related to my choice of programming language. I ended up using python which I had a little bit of experience with but did not fully understand all of the nuances of the language. I learned a lot about the language and if doing things again could now do a better job of implementing much of the design. I would also recommend not attempting to learn a language that hadn’t been used before because it can significantly reduce efficiency in implementation phase.

# Future Work

## Build up framework

At this point of the prototyping phase the project is in a much better spot of understanding what are the necessary pieces to have a fully functional system. Future work would aim to take what has been learned through this iteration of design and implementation and build up a more robust framework for the TFMS to customize to the needs of the users and allow for easier feature upgrades in the future.

## Work On Security

After the structure of the system is redone, I think it would be a good time to reevaluate the systems security. There is room for improvement in user authentication and data integrity. I think that routing all users through a common website and pointing them to a more common type of web server would allow the system to be built on better vetted, more secure systems.

## Dry run

Finally, the system needs to go through an full dry run simulation of a track meet from set up, to execution, to post meet clean up. I think this is an essential step in the evolution of the server to understand what areas are not is good in reality as they looked on paper. This will help the system improve and guide it to being a solution that is viable for prime time