#### **Wegman’s Delivery Management System**

#### **Project/Process Management Plan**

#### **Version 0.3**

#### **09/10/2014**

**Contributors**

**MUSTAFA AL-SALIHI  
CURTIS BURTNER  
MICHAEL YEAPLE   
JEREMY SHULMAN**

#### 

#### 

#### 

#### **Document History**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Revision # | Revision Date | **Description of Change** | **Author** |
| 0.1 | 09/04/14 | Initial Document | Mustafa |
| 0.2 | 9/09/14 | First Version Draft | Curtis, Jeremy, Michael, Mustafa |
| 0.3 | 9/10/14 | First Version Draft | Curtis, Jeremy, Michael, Mustafa |
| 1.0 | 9/22/14 | Version 1.0 | Michael |
| 1.1 | 12/17/14 | Update for Interim Deliverable | Michael |
|  |  |  |  |

1. **Executive Summary** 
   1. Project Overview
   2. In Scope
   3. Out Of Scope
   4. Project Deliverables
   5. Definitions and Acronyms
2. **Risk Management**
3. **Technical Process** 
   1. Methods
   2. Tools & Techniques
   3. Security
   4. Measurements and Metrics
4. **Scheduling**
   1. Resource Allocation
   2. Estimation Techniques
   3. Schedule
   4. Project Tracking

## 1. Executive Summary

### Project Overview

Currently, Wegmans merchant analysts and warehouse employees use a system of spreadsheets and email in order to calculate, predict and communicate the projected amount of inventory to purchase and push to individual stores at the right time. The aim of our product is to both streamline and improve the accuracy of their process by providing them with a web application. This web application will calculate projected inventory using pre-existing algorithms, while also allowing them to modify the inputs to these algorithms. The direct customers of this project are the merchant analysts, warehouse employees, IT department, and store operations employees at Wegmans. Of these customers, all but store operations employees are directly involved in weekly meetings with the project team in order to ensure a quality product.

The initial project schedule has been determined such that there is an initial iteration in which the high level architecture and environment set up will occur. After this iteration, each iteration will occur over the course of three weeks. In the first week, requirements will be prioritized and chosen for the current iteration. In the second week, detailed design for the iteration will occur. In the final week, the design will be implemented and demoed to the customer.

The final product will be an ASP.NET MVC 5 web application written in C#. Since the product is a web application, HTML, CSS, and Javascript are all expected to be incorporated. In addition, Microsoft SQL Server will be used for storing persistent data. The final product will be able to run on a Microsoft Server instance.

Our team aims to make the product as modular and extensible as possible. The project itself is a rather large one. This being the case, we can only consider a portion of the project to be in scope for practical reasons. Therefore, we must make sure to work closely with the Wegmans IT department so that they can continue to expand upon our final product with ease upon the project’s conclusion. Because the quality of our code will be very important to the customer, we have chosen to do test-driven development with pair programming and code reviews prior to committing to the main code branch. The product will be developed iteratively in order to ensure customer visibility and participation.

### In Scope

* A web role which allows users to view and interact with the system will be implemented and delivered.
* An engine role which performs calculations for existing algorithms will be implemented and delivered.
* A means of authenticating users will be implemented and delivered.
* A customizable “portal” for the merchant analysts and warehouse employees to view and interact with the system will be implemented and delivered.
  + Whether or not these are a unified, customizable view or two separate views is to be determined.
* A means of searching for specific items by attributes deemed appropriate (ex. item name, department, etc.) will be implemented and delivered.
  + Please note the properties that are searchable in this feature depend upon the data provided.
* A means of viewing results of the algorithms will be implemented and delivered.
* A means to manipulate the inputs for the existing algorithms will be implemented and delivered.
* A means of sorting items by how statistically anomalous the data is for the item will be implemented and delivered.
* A means of importing data into the system at a single entry point will be implemented and delivered.
* Working with Wegmans developers to import our project into their environment so they expand upon/maintain it upon completion of the project.
* A customizable “portal” for the store operations employees to view future shipments of product to their specific store will be implemented and delivered.

### Out Of Scope

* The ability to add new algorithms through the web interface will not be delivered by this team because we believe it is too large of a feature to fit in alongside our other development responsibilities.
* Localization will not be delivered by this team because Wegmans is based in the East Coast of the United States, and the users of this system are expected to have a primary understanding of English.
* Creation of the data itself and/or any test data will not be delivered by this team because it is expected to be delivered by the company.
* Creation of methods to import the data from other formats than what is originally provided will not be delivered by this team.
  + Importing data will be done from a centralized location with a standard interface so that the Wegmans IT developers may easily add their own methods to import data from other sources.
* Running the web application under a secure protocol (HTTPS) is considered to be out of scope for this project because it is a production environment concern, and the team expects the customer will have its own production environment for the final product.

### Project Deliverables

1. Working versions of the software and any relevant documentation after each iteration.
2. The code repository which houses the application code.
3. A brief, *high* level architecture document which contains technologies to be used.
4. A final design document which is developed iteratively for each release
5. Project website holding all non-proprietary work products and project artifacts maintained in the project account on the se.rit.edu web server. If for some reason, you are hosting project artifacts on another web server, a static mirror image must also be maintained on the department server.
6. Project plan, schedule and process methodology definition prepared by the end of week 4/3 (fall/summer start) of the first term.
7. Tracking report for time/effort worked by each team member and the team aggregate updated on the project website weekly. Tracking report for at least two product/process metrics appropriate to the project and development methodology updated on the project website at least every two weeks.
8. Make interim status and final project presentations. Attend presentations given by other teams and provide constructive feedback on the presentations.
9. Project poster and presentation at "SE Senior Project Day"
10. Project technical report
11. Interim and final team self-assessment
12. Post-mortem curriculum reflection report
13. A CD(s) at the conclusion of the project containing all project artifacts.
14. Each team member completes a Software Engineering Program Senior survey

From: <http://www.se.rit.edu/~swen-561/CourseInformation/Deliverables.html>

### Definitions and Acronyms

A list of terms and agency acronyms used for this project are listed in the “Glossary of Terms” in Section 5 of this document.

## 2. Management Process

### Risk Management

See our separate Risks document. This document will be regularly updated as new risks are identified. In this document, we have the following columns:

* ID
* Title
* Category
* Description
* Probability
* Loss (Weeks)
* Exposure
* Status
* First Indicator
* Mitigation Approach

## 5.1 Risk Identification and Mitigation

Our risks exist in a [separate spreadsheet](https://docs.google.com/spreadsheets/d/1F3WiK4IpPujw-lFhEvMQ1OVupVt3Zh-dvCyYl5WgJEg/edit?usp=sharing). In this spreadsheet, identified risks and their mitigation strategy are listed. Each risk is assigned an ID for tracking purposes and brevity in internal communication. Title is the generalized title of the risk. Description is a more in-depth explanation of the risk. Probability is the chance of the risk occurring, limited to the range 0 (no chance of the risk occurring) to 1 (the risk definitely will occur). Loss is the corresponding potential loss in weeks on the project should the risk occur. Exposure is the probability of occurrence multiplied by the potential loss.

## 3. Technical Process

### Methods

We have chosen Microsoft Team Foundation Server (TFS) as a project management tool. TFS supports user stories, time tracking, and software test plans, and much more. Aside from that, we will have sufficient documentation governing how these functions are supported.

For quality assurance, TFS allows a user to write test scripts and have tracking for them within the current iteration. Since we are using extreme programming, we will also be doing test driven development which makes it so that all tests (unit, integration, regression, etc.) are written before functionality. There will be scripts to run these tests.

Configuration management will follow our requirements document and outlined test plan. It will be seen throughout our tests to ensure good performance and other requirements are met.

Verification and validation will be driven by our pull requests/code reviews. Before we integrate new code with our main branch, we will have it go through a code review where one or two other members of the team who did not work on it will review it. This will ensure that the new code matches what is asked for by the spec and that it functions as expected.

### Tools & Techniques

The programming language chosen for the engine of Simpply is C# with ASP.NET with a SQL database. This requires developers to develop using Visual Studio on a Windows machine. Being a web application, developers will also be using HTML, CSS, and Javascript for the user interface. TFS will be used for our version control so that we have commit history for all of our changes.

Extreme programming is the methodology which will be followed. We will be using TFS for management of the project. We will break the project up into iterations and have a “continuous integration” approach. Due to our methodology, it also follows that we will be pair programming a substantial amount of our code and we will be using test driven development. Pair programming will allow us to write code faster, more efficiently, and with a lesser amount of defects than created when programming alone. It also gives multiple people code expertise which is better than attempting a knowledge dump after a piece of code has been written. Test driven development will allow us to ensure a high level of test coverage before the system is actually developed. It also can be used to guide development as tests will be written to match the spec.

### Security

There are multiple security concerns associated with a web application. These include cross site scripting, remote code execution, cross site request forgery, etc. Our web application will protect against these security concerns. Additionally, user authentication--which will be implemented by Wegmans upon hand-off of the product--will be required to access the system. This authentication will ensure that only authorized users can access the system. This will also provide a means of associating the user type with each individual user. This will govern which view is given to the user upon login.

### Measurement and Metrics

For our project metric, our team has chosen velocity. Velocity is a measurement of the amount of work a team can get done in a single iteration or sprint. A team with high velocity is able to complete more units of work than a team with low velocity. This measurement, together with our ability to estimate, will help us not only decide how much to work on from iteration to iteration, but it will also help us decide if we’re going to be able to complete all major project requirements by the end of the academic year. This measurement can be collected and viewed through TFS, our project management tool. To ensure the measurement is actually meaningful, it will require diligence on behalf of the team to provide accurate estimates and honestly report how many hours have been worked.

To evaluate our product, our team has chosen to measure the cyclomatic complexity of its components. This metric tells us how “complex” a system or subsystem is by measuring the number of linearly independent paths through the program’s code. A highly complex system indicates a lack of cohesion and stability, and a system with low complexity implies the inverse. This metric was selected as a way to help indicate the quality of product being delivered to the customer and reveal areas of the system that may need to be redesigned to reduce complexity. We will be collecting this measurement using tools built into Visual Studio.

To keep our process in check, we plan on keeping track of how often we misestimate a task, and by how wide a margin that misestimation deviated from the number of hours worked. To keep track of this, we will enter estimates before starting a task and track the amount of time spent on that task within TFS. From this, we will be able to see how accurately or inaccurately we estimate tasks, which we will use to improve our estimation process.

Finally, we will use a burndown chart to track our progress through an iteration. Our tool, TFS, is able to quickly and easily display this chart. This measurement will give our customers a good view into how quickly we’re moving in a given iteration.

## 4. Scheduling

### Resource allocation

Developer Time: 1800 Hours

This project will take place over the course of 30 weeks. Our team has 4 developers, and each is expected to invest roughly 15 hours a week. This means our weekly bandwidth across all 4 team members is **60** hours, with the total amount of time spent on the project coming to around **1800** hours of developer time.

Faculty Coach Time: 60 Hours

Our faculty coach will be involved in 1 hour meetings with us and our sponsor every week over the course of 30 weeks. In addition to this, we expect no more than an extra hour - on average - outside of these meetings to review project artifacts and answer questions regarding the project.

Sponsor Time: 150 Hours

Similar to our faculty coach, we expect to meet with at least 1 sponsor representative each week. However, we have so far met with 4 sponsor representatives every week during this project. Based on this, we expect to spend **no more than** 120 hours in 1 hour weekly meetings with 4 sponsors every week for 30 weeks. This amount could be less, depending on how many sponsor representatives come to each meeting. Outside of our weekly meetings, we expect another hour each week from the sponsor to answer emails and perform any assigned action items.

Other Resources Needed:

* Visual Studio 2012
* Microsoft Server VM
* Microsoft SQL Server
* Microsoft Team Foundation Server
* Meeting room access

### Estimation Techniques

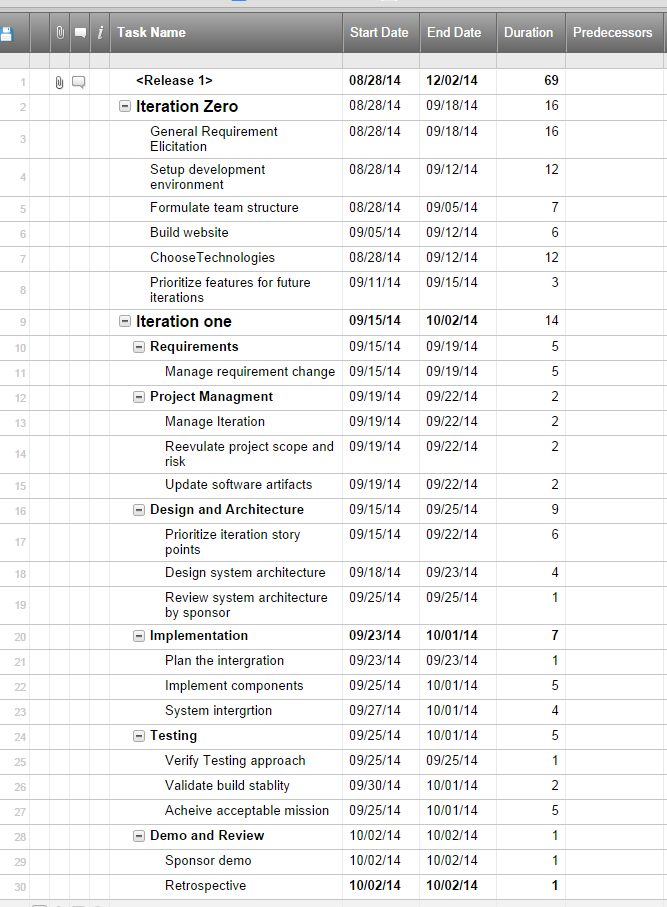
Our team decided to use story points as an estimation technique. Story points are based on the team’s perception of the size of the work. Therefore, the determination of size is based on level of understanding, complexity and the amount of work required. Story points can estimate how much work can be accomplished in an iteration or release which goes along with our process methodology. They also measure scope change.

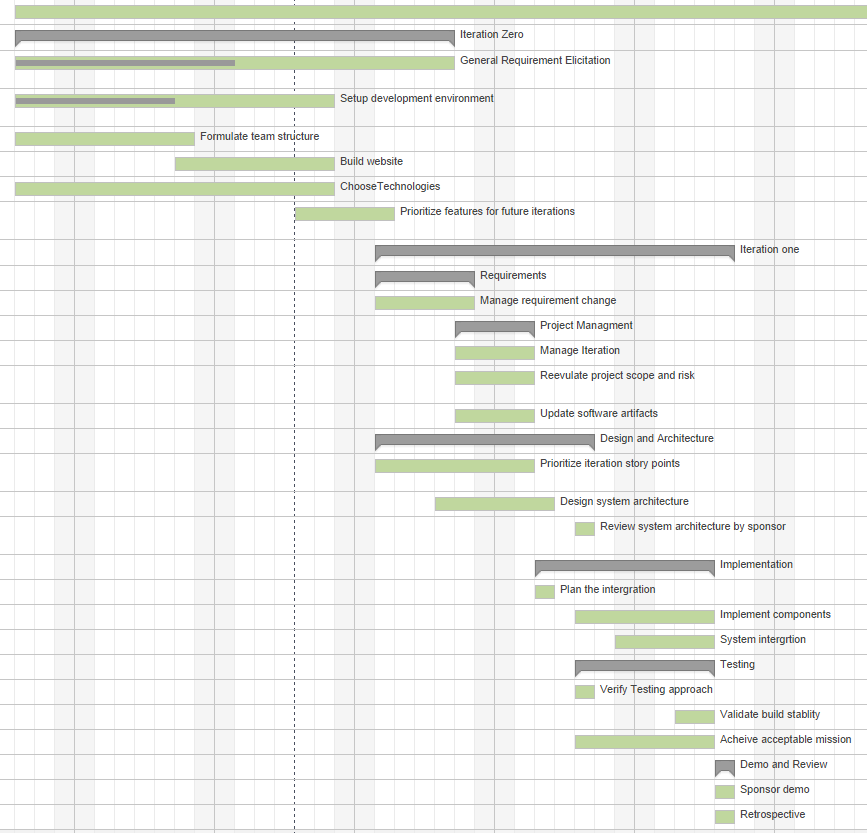
We use the MMRE (Mean Magnitude of Relative Error) to calculate the average estimation accuracy.

*MRE* = \frac{|\text{actual effort} - \text{estimated effort}|}\text{actual effort}

### Schedule

The team has decided to work in four week iterations. We meet with the customer once a week. In the first meeting, we will discuss requirement changes, plan for the iteration by choosing user stories to complete, and gather any necessary information to begin the iteration. In the second meeting, we will review architecture and user experience with the sponsor. In the third meeting, we will gather any necessary information to complete the iteration. In the fourth meeting, we will demo the product. We will provide weekly status updates to the customer, giving the customer clear visibility of project progress and effort. Once per iteration (typically in week 3), we will do a code review with the Wegmans IT team. Below is our detailed schedule[[1]](#footnote-0).





**Project Tracking**

At a high level, the state of the project will be tracked based on the number of hours invested into each of the user stories and the required RIT artifact deliverables. By the end of the project, each user story should either be satisfied by the software solution or determined to be out of scope and each required RIT deliverable should be completed.

At a more granular level, individual tasks will be tracked through the use of TFS, our project tracking tool. We will create and estimate new tasks based on the requirements of the user story or the RIT deliverable being worked on.

1. Please note that this was the original schedule, when we were planning to do three week iterations. The paragraph here has been modified to reflect our four week iteration schedule as of 12/17/2014. [↑](#footnote-ref-0)