UAH Fit Vault

Software Development Plan

6 December 2015

James J. Duggan IV

Glen L. Riden

Whit J. Sisulak

Timothy R. Wilkins

Revision History

Version	*Revision Date*	*Description of Change*	*Author*
0.1	10/17/15	Initial Draft	G. Riden
0.2	10/24/15	Format changes and minor wording changes	J. Duggan
0.3	10/26/15	Merging in the changes from version 0.2 to the previous version to fix “Heading 1” style error. Also added back deleted sections to fill them with N/A if not applicable.	G. Riden
0.4	10/26/15	Added some cross-references throughout the document. Also filled in the not applicable sections with N/A.	G. Riden
0.5	10/26/15	Deleting glossary. Added in more sections.	G. Riden
0.6	11/9/15	Updating with comments from Dr. Kulick.	G. Riden
0.7	11/18/15	Adding page numbers. Updating section 3.	G. Riden
0.8	11/26/15	Adding section 6.X	G. Riden

Contents

<This page intentionally left blank>

1 Scope

1.1 Identification

This document is designed to be applicable to the development stage of the UAH Fit Vault software system. The UAH Fit Vault software system will consist of two tools (data collection and data analysis) which will both be released (version 1.0.0.0.X) to the customer at the end of the development process. There will be no maintenance plan specified by this document, as it is not applicable to the scope.

1.2 System Overview

The UAH Fit Vault software package will be a web application that will accept medical data from users and display the data in a meaningful way. There are two major components to this software. The first is the data collection tool that is used by the users to upload their medical data that is recorded by one of the supported wearable medical devices. There are three different medical devices supported for this project that record various types of data. The data provided by these devices consists of different file formats, and the data is different from device to device. The software will have to determine the contents of each file and how to process them. Due to how long data transfers take to download the data from a device, there may be a need in the future to convert the data from a binary format to another format in order to speed up the process of getting data off the device. The software needs to able to take in files provided by the medical devices process the files, and store the data in a database. The software should have the ability to process multiple files at a time as well as individual files.

The other major component of the web application is the data analysis tools used to analyze the data that is captured from the data collection tool mentioned above. The software needs to perform data analysis over different intervals of time such as one week, one month, etc. There will need to be some way to manage user access to the various medical data that has been inserted into the database that this software will access. Below are some proposed data analysis ideas that can be incorporated into the project.

- Simple Moving Average
- Data correlation discovery between the multiple devices.
- Possibly determine when an individual moves from walking to running or simply being able to identify the activities that were being performed while the data was being captured.

The data analysis possibilities will likely not fully be realized until the project team understands the different types of data that are available. Also, there will need to be collaboration with the customer for additions or changes to the data measurements provided by this software. The web application will have to have different levels of user access which will be defined later in this document.

1.3 Document Overview

The purpose of this document is to detail the software development procedures to be utilized by the Medical Health group. The intended audience for this document is software developers, testers, customers, and any other stakeholders. The software will most likely require additional privacy and security protections due to the nature of the product.

1.4 Relationship to Other Plans

The UAH Fit Vault Software Development Plan is highly dependent on the UAH Fit Vault Configuration Management Plan and Software Test Plan.

2 Referenced Documents

- UAH Fit Vault Configuration Management Plan
- UAH Fit Vault Software Test Plan
- Microsoft Secure PW Guidelines (https://www.microsoft.com/security/pc-security/password-checker.aspx)
- Microsoft C# Coding Conventions (https://msdn.microsoft.com/en-us/library/ff926074.aspx)
- J-STD-016-1995

3 Overview of Required Work

3.1 Requirements and constraints on the system and software to be developed

The requirements and constraints on the system and software to be developed are derived from a feature list from the customer.

3.2 Requirements and constraints on project documentation

N/A

3.3 Position of the project in the system life cycle

The project is currently in the development phase of the system life cycle.

3.4 The selected development/acquisition strategy; any requirements or constraints on it

N/A

3.5 Requirements and constraints on project schedules and resources

The project must be completed by the end of the second semester (May 2016).

3.6 Other requirements and constraints, such as on project security, privacy protection, methods, standards, interdependencies in hardware and software development, etc.

The requirements may be security-critical or privacy-critical. The process for handling these kinds of requirements is detailed in section 4.2.5.

4 Plans for performing general software development activities

4.1 Software development process

The software development process will follow an iterative development process. Each iteration will last approximately two weeks. At the end of each iteration, process will be documented with weekly reports, and plans will be updated accordingly.

The result of this project will be a final official release (version 1.0.0.0.X) to the customer consisting of data collection and data analysis tools.

4.2 General plans for software development

4.2.1 Software development methods

The software development will follow an iterative process. The software will be developed using C# within Visual Studio.

4.2.2 Standards and practices for software products

The software source code written in C# will follow the Microsoft C# Coding Conventions.

Header comments will be located at the beginning of a file and preceding functions. The header comments at the beginning of files will consist of the date modified, author and a brief summary of the updates and/or additions. Header comments preceding functions will be brief when preceding the function prototype and consist of author, date, and a brief summary. A more detailed header comment should be provided preceding the function definition and include details such as author, date and pseudo-code (for complicated algorithms).

4.2.3 Traceability

As requirements are implemented in system design, there will be traceability that links the design item (diagrams, models, test cases, source code, etc.) to the requirement. The traceability will be maintained via a traceability matrix.

4.2.4 Reusable software products

There are no known existing re-useable software products that will used in this project.

4.2.5 Handling of critical requirements

There may be certain privacy and security critical requirements associated with this project due to the medical nature. The software team will analyze each requirement as a whole to determine if it is deemed critical. If a requirement is deemed critical, the costumer will be consulted about how they wish to resolve the issue, including modifying the requirement to change the classification to non-critical.

4.2.6 Computer hardware resource utilization

The SQL database storing the fitness information may need to be monitored to ensure that its size does not exceed virtual machine storage. This is out of the scope of this project, and the maintainer will be responsible for this.

4.2.7 Recording rationale

The development process will be tracked through a GitHub repository. The customer will be given access to the GitHub repository in order to track the history of the project.

4.2.8 Access for acquirer review

The software team will provide access to the data analysis tool and data collection tool and source code as requested by the customer.

5 Plans for performing detailed software development activities

5.1 Project planning and oversight

5.1.1 Software development planning

The software development will be planned throughout the iterative development process. Each iteration will be two weeks long. At the start of each iteration, the desired features to be implemented will be planned. If the software development plan needs to be changed, then those changes will be addressed at the start of an iteration.

5.1.2 Software item test planning

Each software unit test should be developed and executed during the iteration that the unit is implemented.

5.1.3 System test planning

The system test will be performed at the end of the project and during any informal release to the customer prior to the final release.

5.1.4 Software installation planning

The pre-built virtual machine server will be installed at the customer's site when version X.X.0.0.0 is released. During transition, instructions detailing how to build the virtual machine server will be provided to the costumer in a separate document.

5.1.5 Software transition planning

The software will be transitioned to the customer at the end of the project. The end of the project will be signaled by the release of software version 1.0.0.0.X.

5.1.6 Following and updating plans, including the intervals for management review

If plans need to update during development, the need for change will be noted during the iteration that the problem is discovered. The plan will be changed at the start of the next development iteration.

5.2 Establishing a software development environment

5.2.1 Software engineering environment

The software engineering environment will consist of the project team's development computers (personal computers) and the configuration management GitHub repository.

5.2.2 Software test environment

The test environment will consist of the customer's web server to test the web application tool. The web application tool test environment may be stored on a virtual machine for easy back-up and re-load.

5.2.3 Software development library

The system will use the Standard C# library.

5.2.4 Software development files

Software development source files, documents, models and diagrams will be stored on a GitHub repository.

5.2.5 Non-deliverable software

There is no non-delivered software planned for the project.

5.3 System requirements definition

5.3.1 Analysis of user input

The project team met with the customer for two meetings to gather a list of desired features for the system. The project team as a whole translated these features in to requirements for the system. Then, the requirements were sent to the customer for review.

5.3.2 Operational concept

The operational concept is described in the System Overview (section 1.2).

5.3.3 System requirements

After meeting with the customer, system requirements were generated by all members of the project team.

5.4 System design

5.4.1 System-wide design decisions

System-wide design decisions are proposed by one member of the project team and reviewed by the rest of the team. When performing system-wide design, the requirements must be reviewed as a group during a weekly meeting to ensure that all requirements are satisfied by the proposed design decision.

5.4.2 System architectural design

Architectural design decisions are made by the project team members that are associated with the subsystem that is being designed. When performing system architecture design, the requirements must be reviewed to ensure that all requirements are satisfied by the proposed architecture design.

5.5 Software requirements definition

The software requirements were defined via the same method that the system requirements were defined. See section 5.3.

5.6 Software design

5.6.1 Software item-wide design decisions

The section shall follow the same procedures as 5.4.1 (System-wide design decisions).

5.6.2 Software item architectural design

The section shall follow the same procedures as 5.4.2 (System architectural design).

5.6.3 Software item detailed design

The section shall follow the same procedures as 5.4.2 (System architectural design).

5.7 Software implementation and unit testing

5.7.1 Software implementation

Software implementation will consist of a master trunk having feature branches and hotfixes being merged in to it throughout the development process. The UAH Fit Vault Configuration Management Plan details the software process further.

5.7.2 Preparing for unit testing

All modules should be evaluated by the author(s) to determine if unit testing should be performed on the module. All modules requiring unit testing should be designed during the software development process to support unit testing.

5.7.3 Performing unit testing

Unit testing will be performed after a module has been fully developed. If the module is a large module, unit testing may also be performed throughout development.

5.7.4 Revision and retesting

If a module fails unit tests, then the unit tests should be rerun on the module or the tests will be re-evaluated to ensure expected behavior is correct.

5.7.5 Analyzing and recording unit test results

The UAH Fit Vault Software Test Plan details the analysis of the unit testing and the method of how the unit test results are saved.

5.8 Unit integration and testing

Section 5.X of the UAH Fit Vault Software Test Plan details the unit integration and unit testing processes used during development.

5.9 Software item qualification testing

Section 5.X of the UAH Fit Vault Software Test Plan details the software item qualification testing processes used during development.

5.10 Software/hardware item integration and testing

Section 5.X of the UAH Fit Vault Software Test Plan details the software/hardware item integration and testing processes used during development.

5.11 System qualification testing

Section 5.X of the UAH Fit Vault Software Test Plan details the system qualification testing processes used during development.

5.12 Preparing for software use

5.12.1 Preparing the executable software

The executable software will be prepared as a pre-configuration virtual machine. The customer will need to execute the virtual machine to deploy the system.

5.12.2 Preparing version descriptions for user sites

Since there is only one release planned (1.0.0.0.X) to be released to the customer, there will be a single version description provided.

5.12.3 Preparing user manuals

A user manual will be provided to the customer after the software has officially been released.

5.12.4 Installation at user sites

The software/system will be installed on the customer's site when the software is complete.

5.13 Preparing for software transition

5.13.1 Preparing the executable software

The executable software will be prepared as a pre-configuration virtual machine. The customer will need to execute the virtual machine to deploy the system. During transition, instructions detailing how to build the virtual machine server will be provided to the costumer in a separate document.

5.13.2 Preparing source files

Source files for the data collection tool will be provided to the customer at the end of development via a cabinet file (Windows or Winzip or 7zip). Source code for the web analysis tool will be provided to the customer in a separate cabinet file (Windows or Winzip or 7zip).

5.13.3 Preparing version descriptions for the maintenance site

N/A

5.13.4 Preparing the "as built" software item design and other software maintenance information

Since maintenance is not in the scope of this project, it will be up to the customer to determine a maintenance plan.

5.13.5 Updating the system design description

If the system design description needs to be updated, the request-for-change will be noted during the current iteration of development that the problem is discovered. The change will be address at the beginning of the subsequent development iteration.

5.13.6 Updating the software requirements specification

The Software Requirements Specification will be finalized after 6 January, 2016. The Software Requirements Specification will be delivered to the customer after it is finalized.

5.13.7 Updating the system/subsystem specification

The system/subsystem specification will be updated at the start of each development iteration if necessary.

5.13.8 Preparing maintenance manuals

Since maintenance is not in the scope of this project, it will be up to the customer to determine a maintenance plan.

5.13.9 Transition to the designated maintenance site

Since maintenance is not in the scope of this project, it will be up to the customer to determine a maintenance plan. There is no plan to transition the system to any other site than the deployment system.

5.14 Software configuration management

The configuration management for the development process is specified in the UAH Fit Vault Configuration Management plan.

5.15 Software product evaluation

5.15.1 In-process and final software product evaluations

The system may be evaluated by the customer during the software development process when requested. Otherwise, the final software evaluation will occur during time of formal qualification testing.

5.15.2 Software product evaluation records, including items to be recorded

A record of in-process software evaluations will be kept, including the date of when the evaluation occurred and comments arising from the evaluation.

5.15.3 Independence in software product evaluation

The customer will be responsible for the independent evaluation for the software.

5.16 Software quality assurance

The UAH Fit Vault Configuration Management Plan details the quality assurance processes used during development.

5.16.1 Software quality assurance evaluations

Once a month, two members of the team will review to ensure all processes were followed during development.

5.16.2 Software quality assurance records, including items to be recorded

If a problem is found with the quality assurance process, the problem will be recorded and documented in a quality assurance log.

5.16.3 Independence in software quality assurance

The two members of the team performing the quality assurance evaluation will be responsible for being independent.

5.17 Corrective Action

N/A

5.18 Joint technical and management reviews

During the development process, the customer will be invited to participate in a review once a month.

5.19 Risk management

All risks discovered during a development iteration will be discussed at the end of the development cycle, assessed, and a mitigation plan will be discussed if needed.

5.20 Software management indicators

N/A

5.21 Administrative security and privacy protection

The software source code will be restricted to be viewable by only the project team, the customer, a maintainer when the software is transitioning in to maintenance, and the professor of the class.

5.22 Managing subcontractors

N/A

5.23 Interfacing with Software IV&V agents

N/A

5.24 Coordinating with associate developers

N/A

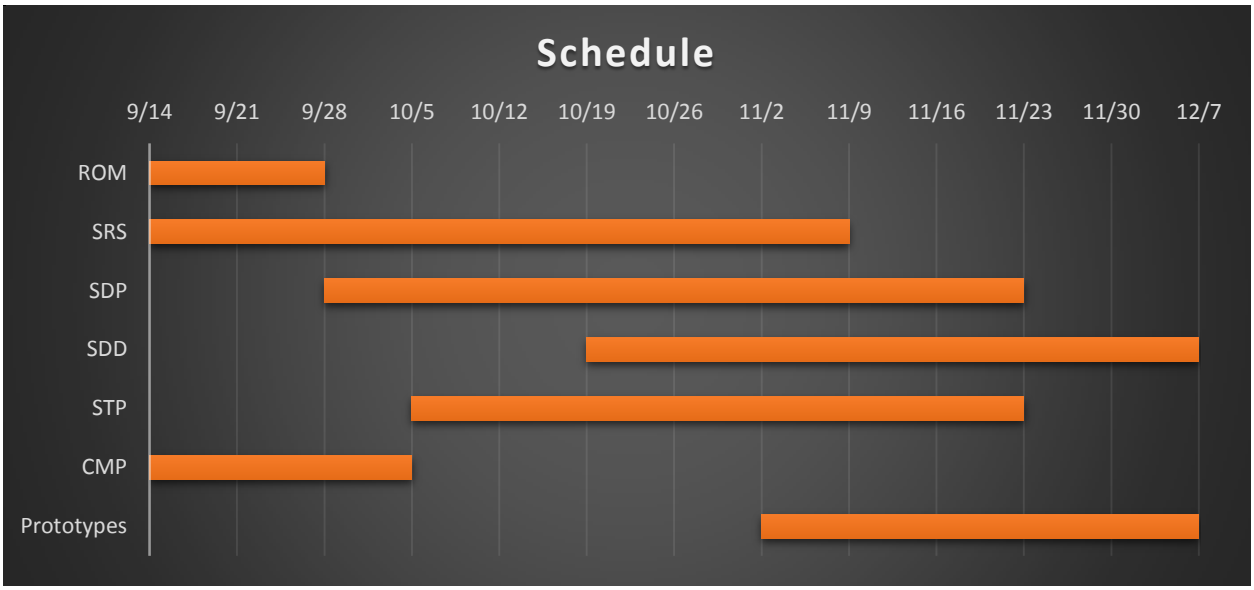
5.25 Project process improvement

N/A

6 Schedules and activity network

6.1 Schedules

The following is the schedule of the first semester of the CPE656/658 course sequence. This first semester is comprised of mostly document creation.



The second semester consists of writing and implementing code and testing. The final deliverable will be at the end of the second semester (May 2016).

6.2 Activity Network

The above schedule shows the dependencies among documents created (earlier products are necessary for later products). All of the products from the second semester are dependent upon these first semester deliverables.

7 Project Organization and Resources

7.1 Project Organization

N/A

7.2 Project Resources

N/A

8 Notes

N/A

9 Annexes

N/A