Project Plan

Trillium Health Bluetooth Event Tracker

Team *: Gimball3000 - Danielle Neuberger, Randy Goodman, Anshul Kapoor, Tyler Schoen*
Sponsor*: AJ Blythe*
Advisor*: Rick Weil*

Table of Contents

1. Overview

Trillium Health Bluetooth Event Tracker is web and mobile application to be used for tracking bikers during the annual Trillium Aids Red Ribbon Ride racing event. This is a week long charity racing event held around the finger lakes region, NY where bikers race to raise funds for aids research. The core functionality of this application relies on bluetooth low energy devices (beacons) that enables race staff to automatically know when bikers pass by the checkpoints. The application also allows support staff to update the status of each biker (when help is needed). Another key functionality is to store offline data regarding each biker and sync it with the servers, once stable internet connection is available.

The team has meet with the customer and the agreed solution is an iOS application integrated with the bluetooth devices, as well as an accompanying web application, both of which serve different purposes. The web application will focus on user data from administrative perspective, and any participant features like personal pages and social integration, whereas the iOS application will be carried by race staff and will be used for in-race features, such as to receive, log, and display participant updates. The application will also be generalized to allow for other types of races besides the initially agreed upon event race tracking.

This app will be supported on iOS platform, while a web version will be available for race administrators.

The end customers of the application are the management staff who help run the event. This includes SAG (Support, Assistance, Gear) crew, admin staff and others who help manage the event. The staff can communicate with each others using the application, and also check on the position of bikers. Competitors will also be able to see various statistics regarding their trip and share this information with potential donors.

The entirety of the project in an agreed-upon scope will be completed within the time frame of RIT's fall and spring semesters, which run from August until May. The team will schedule features and milestones appropriately with signoff from the project sponsor in order to complete the project during the time frame, and will need to account appropriately for breaks, days off, and risks such as team members becoming sick, all of which affect schedule.

The team consists of diverse skill sets and following is a breakdown of the roles:
Randy Goodman: Database and Server API
Anshul Kapoor: Server API & iOS dev.
Danielle Neuberger: Project Management
Tyler Schoen: iOS & front-end dev.

Development responsibility will be shared among all team members, but team members will have some specialization according to the roles listed above.

2. Document Organization

This document is meant to provide a detailed view of the various aspects of the Trillium Bluetooth Event Tracking project. It covers everything from project scope to risks, from schedule to metrics, and many other helpful aspects.

Certain sections of this document use Priority values to convey importance of certain features/goals. The priority values are on a scale of P1 to P3. An explanation of the priority values follows:

- P1 is the topmost priority and is absolutely necessary for a viable product. Without these items, the product should not be released.
- P2 items are important and impact quality of the product, but the product would still be shippable without items in this category.
- P3 items are nice to have but can be dropped if necessary. They are not required for a quality shippable product. This category includes enhancements.

Other tables in some sections may use different measurements related to their specific purpose or topic (i.e. the risk scoring table contains impact and probability values). The values for those tables will be defined in a description directly above the table.

3. Goals & Scope

Goals provide the primary objectives for the project and help define the scope. The following two sections specify this project's prioritized goals and a series non-goals with explanations, in order to clarify scope, intentions, and direction of the project.

3.1. Goals

#	Goal	Priority
1	Ensure participants' safety during their activity	P1
1.1	Provide biker check in/check out	P1
1.2	Provide details of biker's status	P1
1.3	Estimate biker location/arrival information	P2
2	Provide beacon tracking in a standalone fashion, and capability to share that information over cellular networks as available	P1
3	Support concurrent events	P1
4	Allow staff to efficiently communicate with each other	P1
5	Enable the system to be configured for a variety of different event types	P2
6	Provide means for event promotion	P3

3.2. Non-Goals

Defining non-goals clarifies the scope of the project by specifying attributes or functionality that are not in the scope of the project. The following table defines these non-goals and provides explanation as to why they are excluded for the project.

#	Non-Goal	Reasoning
1	Virtual races	The main goal of the product is ensuring participant safety on a physical, centrally located racecourse.
2	Continuous tracking	Beacons are tracked only when proximal to a tracking device.
3	Fundraising Focus	This is not a fundraising event!
4	Route Creation	It's not meant to alter race routes (MapMyRide can do that), and it would be a separate project in its own.
5	Race Timing	Its meant to estimate the ETAs of racers, not time them.

4. Technical Process

4.1. Methodology

In selecting a software development methodology, our team considered the gamut of options. Our thoughts on some of the methodologies are enumerated below:

- Waterfall - May not be adaptive enough for our needs.
    - Modified Waterfall - Phases can overlap, but still not much room to add additional features or modify existing work.
- Evolutionary Prototyping - Requirements are pretty well defined, enough so that evolutionary prototyping would be a waste of team effort.
- Iterative/Incremental Model- Viable choice given the how the requirements are broken down into set objectives. Potentially may not be adaptive enough for the beginning stages of the development cycle since rigid and phases cannot overlap.
- RUP - May be too heavyweight since most of the requirements and objectives are already laid out. However, this methodology has the design completed upfront and then an iterative approach integrated later. This may not be light enough to adjust to early stage changes.
- Agile - May be unnecessarily light, requirements have been laid out at the beginning of the project and should not change all that much.
- **Spiral** - Seems like the right approach for a project with such requirements, but may be a little heavy on the risk analysis and risk management process. However, that may not be a con.
- V-model - Emphasises on testing, which isn't a con. However, it may not be adaptable as we need in the later stages of development.
- XP - Unnecessary and too lightweight for the needs of this project.

From the above discussion it can be seen how each methodology would fit into this specific project, our team ended up deciding to use the Spiral Methodology. The Spiral Methodology is a methodology focused on risk management that combines elements of iterative and prototype models. This will allow us to plan, analyze, and implement individual objectives laid out and gathered through the project proposal. Some pros of spiral are that it is relatively flexible, especially when compared to more rigid model such as waterfall. It also has risk management built-in, and estimations become more accurate as the project moves forward. This will allow us to easily plan for the requirements laid out for us as well as quickly gather unforeseen requirements and adapt to them. Additionally, prototypes are developed in each iteration so there is high visibility and customer feedback can be easily gathered and incorporated.

However, there are also some cons in choosing Spiral that we should take into account. Reviewing and evaluating the project after iterations, requires experience and expertise. It's also worth noting that in order for Spiral to be effective it is imperative that all rules of the methodology are followed, this can be tough throughout the lifespan of the project. Also, the amount of documentation required at the beginning and intermediate stages of the project is

fairly large compared to some other models. This can make project management complicated if this documentation is not managed properly.

The team will implement the Spiral Methodology in month-long iterations, which will provide for approximately four iterations per semester. Given that each iteration has four stages, the team has agreed month-long iterations should provide enough time to go through each stage and not feel rushed. This schedule will be evident and documented in the project schedule, which will be managed in the team Google Drive.

4.2. Tools and Techniques

- Version control - git
    - This one was an obvious choice as the majority of the team has experience with git, this also allows us to easily host our code in a private repo on Github.

- Project information website - Github Pages
    - Coupled with git/Github, Github pages makes it extremely easy to manage a project website without all the hassle of handling HTML and CSS.
    - The Github Pages site will be periodically migrated to the official team website on the SE domain - http://www.se.rit.edu/~gimball3000/

- Scheduling software - Gantter
    - Gantter is a project management application that integrates nicely with Google Drive, which was one of the primary reasons for selecting it over Microsoft Project. If necessary, it can also import and export to MS Project as well. It contains tools for schedule management including a Gantt chart and WBS, as well as resource management. It also has tools for team member/project calendars and risk management, but our team will not be utilizing those because they are not robust enough.

- Team Communication - Slack
    - Slack is a great tool for convenient, professional team communication that also provides exceptionally useful tools for project management and development as well as service integration.

- Task Management - Trello
    - Trello is a simple and easy to use task management tool that integrates with Slack. This will provide the team with a central location to track, update, and manage tasks for each iteration of the project.

- Documentation Storage/Management - Google Drive
    - Google Drive is the most convenient documentation storage service used among the team. Google Drive allows for easy document creation, organization, and history , which exactly what we want for managing our project documentation. A

primary driver for using Google Drive/Google Docs is the capability to have multiple team members editing files in real-time and communicating using the chat feature.

- Continuous Integration - Jenkins
    - Jenkins will be used for Continuous Integration because it has such large popularity in the development community that many team members have used it before. It also integrates with Slack for updates.

4.3. Internal Artifacts

As part of our decision to use the Spiral Methodology, our team will be creating the following internal artifacts:

- SRS (Software Requirements Specification) - outlines the requirements; will be kept up to date and maintained
- Prototypes - completed each iteration at the end of the risks phase and used to identify and resolve critical risks before moving on to the development phase
- Software Architecture Document - specifies the software architecture including physical and logical elements and their relationships, as well as any COTS (Commercial Off-The-Shelf), open source, or reusable software components. Will incorporate some diagrams.

5. Deliverables

Deliverables are derived from the required senior project deliverables, project-specific requirements, and deliverables mandated by our chosen process. Reference section 7 for process details, and section 5 for information related to schedule of deliverables and the project overall.

At a high level, major deliverables for the project include:

- **Project Planning document (this document)** - Will contain information including but not limited to overview, goals, scope definition, risks, scheduling, and process methodology definition.
- **Project website** - This will contain information about the project, including all non-proprietary work products and artifacts.
- **Domain model** - Will be used to describe and demonstrate an understanding the application domain.
- **Weekly four-up charts** - These will be reviewed at the beginning of each weekly meeting and will be used to gauge status and project progress.
- **Time tracking** - Time/effort worked will be tracked by each team member and then aggregated weekly to be recorded on the project website.
- **Select process/product metric tracking** - Every two weeks or team will track and provide updates on the website for at least two process/product metrics. This will provide a progressive view of the project and allow us to concentrate on improvement.

- **Interim and final project presentations** - Will be used to provide midpoint and final status/progress information, as well as to explain and receive feedback on project approach and execution. Team members will additionally attend other teams' presentations and provide feedback.
- **Final project poster and presentation** - This formal presentation will be used to present our work once completed to raise awareness and receive feedback from others.
- **Technical report** - Will present a comprehensive technical view of the project.
- **Interim and final team self-assessments** - These will be used to gauge progress and participation of each team member.
- **Post-mortem reflection report** - Will provide a means to reflect on the project, team's strength and weaknesses.
- **CD containing project artifacts** - Will contain artifacts related to the project to be delivered to the customer.
- **Individual senior project survey** - Used by the software engineering faculty for future senior projects.

From Spiral Model:

- **SRS (Software Requirements Specification)** - outlines the requirements; will be kept up to date and maintained
- **Prototypes** - completed each iteration at the end of the risks phase and used to identify and resolve critical risks before moving on to the development phase
- **Software Architecture Document** - specifies the software architecture including physical and logical elements and their relationships, as well as any COTS (Commercial Off-The-Shelf), open source, or reusable software components. Will incorporate some diagrams.

6. Risk Management

Risks are managed in a separately maintained Google Sheets file on the project Google Drive, which can be found here. In the file each risk has a succinct name along with the following attributes.

- **Probability** is a percentage value from 0-1, which is the chance of the risk occurring.
- **Impact** three value options (low, medium, or high), which corresponds to the effect (of the risk) on the project overall.
    - Low impact : if the risk does occur, the impact is minimal and manageable.
    - Medium impact : the occurrence of the risk will not stop progress completely, but may affect the performance in future.
    - High impact : the project would be hugely affected if the risk occurred, and it would result in an un-shippable product.
- **Exposure** value is a number calculated using the probability multiplied by the impact value (the impact values of low impact is a value of 1, medium is 5, and high is 10). Using risk exposure, the risks can be prioritized so the team can focus on more important risks.

- **Classification** of risks for the table are drawn from a suggestions from a few academic papers, including (2.1) , and (2.2). The resulting category options for our table are: requirements, costs, quality, and scheduling. If a risk fits into multiple categories, its category will be the one it fits into best.
- **Owner** associated with each risk is the assignee responsible for ensuring the risk is prevented and then mitigated and managed if necessary.
- **Mitigation plan** for each risk details our team's plan to prevent the risk from occurring.
- **Management plan** for each risk explains our team's for minimizing impact of the risk should it occur.
- **Status** is the current status of the risk. Options include active, inactive, mitigated, managed.

7. Scheduling and Estimates

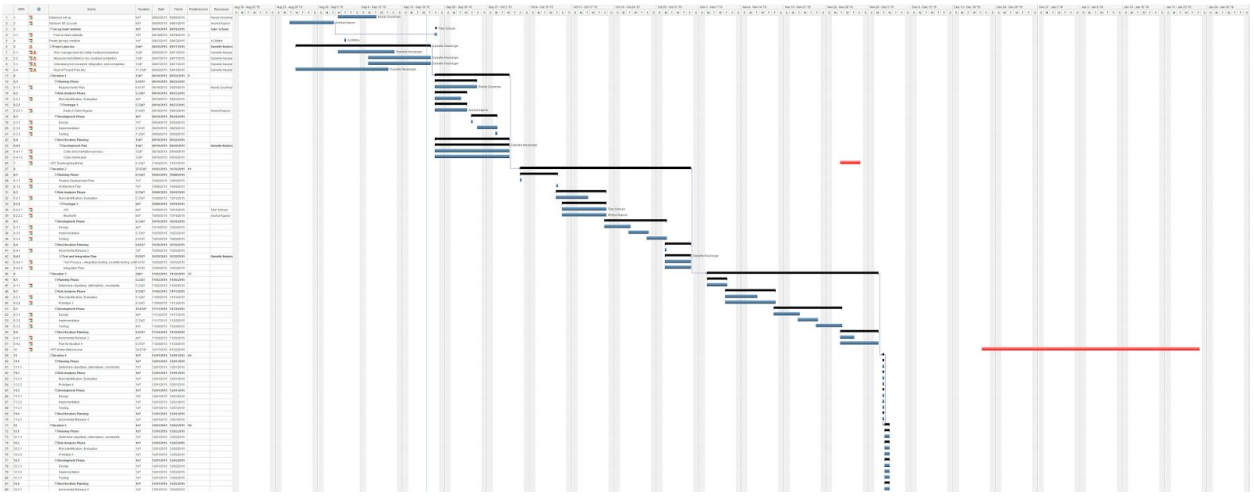
Schedule and resources will be managed using Gantter, a project management tool that integrates with Google Drive. The Gantter project for the team can be found in the team's Google Drive, as well as here.

7.1. Project Schedule

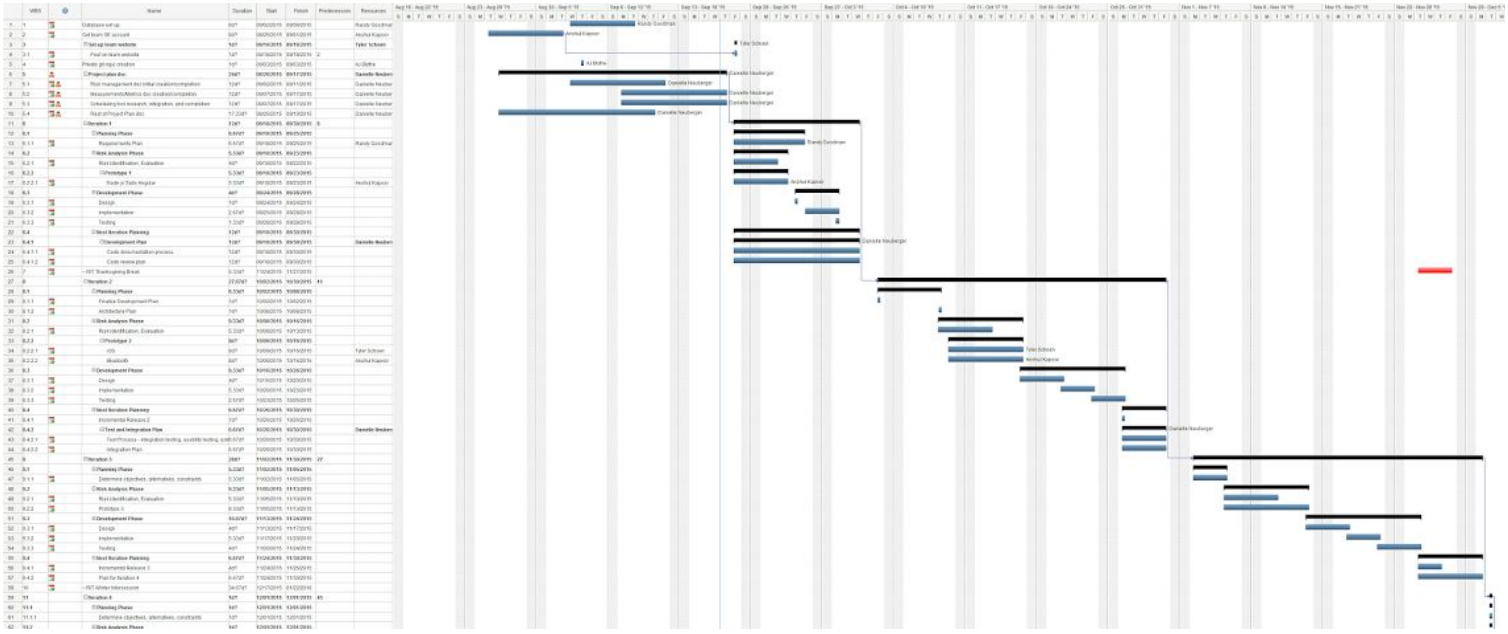
Overall project schedule is framed within the RIT fall and spring semesters time frame, accounting for breaks and days off. The schedule also attempts to account for risks and unexpected changes. It has been created around the Spiral Methodology of development (see the Technical Process section) and includes deliverables and schedule items related to the methodology. For an example and clarification, see the initial static schedule screenshot below.

The project schedule will be managed within Gantter.

A static initial schedule is below for reference. This includes only some initial tasks and first couple iterations since following iterations depend on the requirements document being made in the first iteration.
Because the schedule is so detailed, it may not be readable as depicted here. In order to see more detail, please reference the source Gantter schedule file in the team's Google Drive mentioned before, which can be found here.



The Gantt chart close-up:



7.2. Work Breakdown Structure

The Work Breakdown Structure, or WBS, shows a hierarchical view of high level tasks in the project and highlights their relationships to one another.

The WBS will be managed in Gantter, but a static initial WBS created for version 1 of this Project Plan is below for reference. This again only includes the initial tasks created and the first couple iterations, since defining following iterations depends on creation of the requirements document.

	WBS		Name	Duration	Start	Finish	Predecessors	Resources
1	1		Database set up	0d?	09/02/2015	09/09/2015		Randy Goodman
2	2		Get team IDE account	0d?	09/09/2015	09/09/2015		Anshul Kapoor
3	3		Set up team website	5d?	09/10/2015	09/16/2015		Tyler Schoen
4	3.1		Post on team website	1d?	09/16/2015	09/16/2015	2	
5	4		Private git repo creation	1d?	09/03/2015	09/03/2015		Ai Ohno
6	5		Project plan doc	24d?	08/26/2015	09/17/2015		Danelle Reuben
7	5.1		Risk management doc initial creation/completion	12d?	09/02/2015	09/11/2015		Danelle Reuben
8	5.2		Measurements/Metrics doc creation/completion	12d?	09/07/2015	09/17/2015		Danelle Reuben
9	5.3		Scheduling tool research, integration, and completion	12d?	09/07/2015	09/17/2015		Danelle Reuben
10	5.4		Rest of Project Plan doc	17.33d?	08/26/2015	09/16/2015		Danelle Reuben
11	6		Iteration 1	12d?	09/16/2015	09/30/2015	5	
12	6.1		Planning Phase	6.67d?	09/16/2015	09/23/2015		
13	6.1.1		Requirements Plan	6.67d?	09/16/2015	09/23/2015		Randy Goodman
14	6.2		Risk Analysis Phase	5.33d?	09/16/2015	09/23/2015		
15	6.2.1		Risk Identification, Evaluation	4d?	09/16/2015	09/22/2015		
16	6.2.2		Prototype 1	5.33d?	09/16/2015	09/23/2015		
17	6.2.2.1		Node js Todo Angular	5.33d?	09/16/2015	09/23/2015		Anshul Kapoor
18	6.3		Development Phase	4d?	09/24/2015	09/30/2015		
19	6.3.1		Design	1d?	09/24/2015	09/24/2015		
20	6.3.2		Implementation	2.67d?	09/25/2015	09/29/2015		
21	6.3.3		Testing	1.33d?	09/30/2015	09/30/2015		
22	6.4		Next Iteration Planning	12d?	09/16/2015	09/30/2015		
23	6.4.1		Development Plan	12d?	09/16/2015	09/30/2015		Danelle Reuben
24	6.4.1.1		Code documentation process	12d?	09/16/2015	09/30/2015		
25	6.4.1.2		Code review plan	12d?	09/16/2015	09/30/2015		
26	7		IRT Thanksgiving Break	5.33d?	11/24/2015	11/30/2015		
27	8		Iteration 2	27.67d?	10/02/2015	10/30/2015	11	
28	8.1		Planning Phase	6.33d?	10/02/2015	10/08/2015		
29	8.1.1		Finalize Development Plan	1d?	10/02/2015	10/02/2015		
30	8.1.2		Architecture Plan	1d?	10/02/2015	10/02/2015		
31	8.2		Risk Analysis Phase	5.33d?	10/02/2015	10/16/2015		
32	8.2.1		Risk Identification, Evaluation	5.33d?	10/02/2015	10/12/2015		
33	8.2.2		Prototype 2	8d?	10/06/2015	10/16/2015		
34	8.2.2.1		iOS	8d?	10/06/2015	10/15/2015		Tyler Schoen
35	8.2.2.2		Bluetooth	8d?	10/06/2015	10/16/2015		Anshul Kapoor
36	8.3		Development Phase	5.33d?	10/19/2015	10/26/2015		
37	8.3.1		Design	4d?	10/19/2015	10/22/2015		
38	8.3.2		Implementation	5.33d?	10/20/2015	10/27/2015		
39	8.3.3		Testing	2.33d?	10/26/2015	10/29/2015		
40	8.4		Next Iteration Planning	5.67d?	10/26/2015	10/30/2015		
41	8.4.1		Incremental Release 2	1d?	10/26/2015	10/26/2015		
42	8.4.2		Test and Integration Plan	5.67d?	10/26/2015	10/30/2015		Danelle Reuben
43	8.4.2.1		Test Process – integration testing, usability testing, scalability	5.67d?	10/26/2015	10/30/2015		
44	8.4.2.2		Integration Plan	5.67d?	10/26/2015	10/30/2015		
45	9		Iteration 3	26d?	11/02/2015	11/30/2015	27	
46	9.1		Planning Phase	5.33d?	11/02/2015	11/09/2015		
47	9.1.1		Determine objectives, alternatives, constraints	5.33d?	11/02/2015	11/09/2015		
48	9.2		Risk Analysis Phase	5.33d?	11/02/2015	11/13/2015		
49	9.2.1		Risk Identification, Evaluation	5.33d?	11/05/2015	11/13/2015		
50	9.2.2		Prototype 3	5.33d?	11/05/2015	11/13/2015		
51	9.3		Development Phase	16.67d?	11/13/2015	11/30/2015		
52	9.3.1		Design	4d?	11/13/2015	11/17/2015		
53	9.3.2		Implementation	5.33d?	11/17/2015	11/23/2015		
54	9.3.3		Testing	4d?	11/23/2015	11/26/2015		
55	9.4		Next Iteration Planning	5.67d?	11/24/2015	11/30/2015		
56	9.4.1		Incremental Release 3	4d?	11/24/2015	11/25/2015		
57	9.4.2		Plan for Iteration 4	5.67d?	11/24/2015	11/30/2015		
58	10		IRT Winter Intersession	34.67d?	12/14/2015	01/29/2016		
59	11		Iteration 4	5d?	12/01/2015	12/07/2015	45	
60	11.1		Planning Phase	5d?	12/01/2015	12/07/2015		
61	11.1.1		Determine objectives, alternatives, constraints	1d?	12/01/2015	12/01/2015		
62	11.2		Risk Analysis Phase	5d?	12/01/2015	12/07/2015		
63	11.2.1		Risk Identification, Evaluation	1d?	12/01/2015	12/01/2015		
64	11.2.2		Prototype 4	1d?	12/01/2015	12/01/2015		
65	11.3		Development Phase	5d?	12/01/2015	12/07/2015		
66	11.3.1		Design	1d?	12/01/2015	12/01/2015		
67	11.3.2		Implementation	1d?	12/01/2015	12/01/2015		
68	11.3.3		Testing	1d?	12/01/2015	12/01/2015		
69	11.4		Next Iteration Planning	5d?	12/01/2015	12/07/2015		
70	11.4.1		Incremental Release 4	1d?	12/01/2015	12/01/2015		
71	12		Iteration 5	5d?	12/01/2015	12/07/2015	59	
72	12.1		Planning Phase	5d?	12/01/2015	12/07/2015		
73	12.1.1		Determine objectives, alternatives, constraints	1d?	12/01/2015	12/01/2015		
74	12.2		Risk Analysis Phase	5d?	12/01/2015	12/07/2015		
75	12.2.1		Risk Identification, Evaluation	1d?	12/01/2015	12/01/2015		
76	12.2.2		Prototype 5	1d?	12/01/2015	12/01/2015		
77	12.3		Development Phase	5d?	12/01/2015	12/07/2015		
78	12.3.1		Design	1d?	12/01/2015	12/01/2015		
79	12.3.2		Implementation	1d?	12/01/2015	12/01/2015		
80	12.3.3		Testing	1d?	12/01/2015	12/01/2015		
81	12.4		Next Iteration Planning	5d?	12/01/2015	12/07/2015		
82	12.4.1		Incremental Release 5	1d?	12/01/2015	12/01/2015		

7.3. Resource Allocation

All resources will be managed in Gantter.

Available personnel resources are limited to the current team members, which include Anshul Kapoor, Danielle Neuberger, Tyler Schoen, and Randy Goodman.

Hardware resources include the limited number of bluetooth beacons the project sponsor has brought to share with the team for testing purposes, as well as any iPads or mobile devices the sponsor provides for testing. The team will additionally be using our own personal laptop/desktops and mobile devices for testing purposes.

Software resources include licenses provided by the sponsor, such as the private Github repository.

7.4. Estimates

Estimations will be done in Gantter and in the Requirements document created after spiral model iteration 1. Estimations will include team effort/time estimation as well as product size (function points).

7.5. Tracking and Schedule Changes

All tracking and schedule changes will be managed within the Gantter project.

8. Measurements and Metrics

Measurements and metrics are recorded and managed in a separate Google Sheets file in our project's Google Drive, which can be found here.

Measurements are data points that can be gathered from the project in order to create metrics. Measurements by themselves have no real value and require metrics in order to be given significance. Each measurement in the file has a measurement name, description, and significance.

Metrics use measurements and make them meaningful by providing information that is useful, actionable, provides an indication of something, or shows some trend that can be useful for predictions. Each metric in the file has a metric name, associated measurements, and an explanation of its significance.

9. Appendix

9.1. Project Plan Doc Additional Info/Resources

" Project Plan Guidelines

*A project plan is the most important, yet often neglected artifact for the typical software development project. A project plan is a sufficiently complete, **professional** document for communicating information to software engineers and management needed to understand what the project entails, how it will be produced and controlled and what the effort and schedule estimates are for the project. The project risks, quality focus, and support needs are specified as well.*

References:

- *Rapid Development by Steve McConnell (available on Books 24x7) see Chapter Seven for background on software development life-cycle models.*
- *Construx Software templates & examples (Steve McConnell's company) see "Engineering Management Section". Note that you will need to create a free login for this site. There are many very useful document templates and checklists available here.*
- *Planning and tracking spreadsheet*
- *Risk management spreadsheet*
- *As part of the Microsoft Academic Alliance, you may download a free copy of Microsoft Project. "*

9.2. References

1. Methodologies
   - 1.1. http://www.slideshare.net/RiantSoft123/different-types-of-software-development-model
   - 1.2. https://melsatar.wordpress.com/2012/03/15/software-development-life-cycle-models-and-methodologies/
   - 1.3. http://www.ijcsi.org/papers/7-5-94-101.pdf
   - 1.4. https://docs.google.com/viewer?url=http%3A%2F%2Fwww2.engr.arizona.edu%2F~ece473%2Freadings%2F2-Comparison%2520of%2520Software%2520Development%2520Methodologies.doc
   - 1.5. http://www.ijarcsse.com/docs/papers/May2012/Volum2_issue5/V2I500405.pdf
2. Risks
   - 2.1. http://www.academia.edu/573832/Classification_and_Analysis_of_Risks_in_Software_Engineering
   - 2.2. http://project-management.com/understanding-the-4-types-of-risks-involved-in-project-management/
   - 2.3. http://www.mitre.org/publications/systems-engineering-guide/acquisition-systems-engineering/risk-management/risk-impact-assessment-and-prioritization
   - 2.4. http://www.sersc.org/journals/IJSEIA/vol7_no1_2013/5.pdf