**Portland State University**

**Electrical and Computer Engineering Department**

11

**Capstone 2011**

**TIU Tracking Project Proposal**

**Sponsor: Intel**

**Advisor: Prof. Robert Daasch**

**Members: Dung Le**

**Tri Truong**

**Lynh Pham**

**Man Hoang**

**Daniel Ferguson**

**Version History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version** | **Implemented**  **By** | **Revision**  **Date** | **Approved**  **By** | **Approval**  **Date** | **Reason** |
| *1.0* | *Lynh Pham* | *1/22/2011* | *Prof. Daasch* | *01/25/2011* | *Proposal outline* |
| *2.0* | *Dung Le*  *Tri Truong*  *Lynh Pham*  *Man Hoang*  *Daniel Ferguson* | *2/06/2011* |  |  | *Draft Proposal* |
| *3.0* | *Dung Le*  *Tri Truong*  *Lynh Pham*  *Man Hoang*  *Daniel Ferguson* | *2/15/2011* |  |  | *Proposal* |
| *3.2* | *Dung Le*  *Tri Truong*  *Lynh Pham*  *Man Hoang*  *Daniel Ferguson* | *2/17/2011* | *Prof. Daasch* | *02/17/2011* | *Modify as recommended by Prof. Daasch* |
| *3.3* | *Man Hoang*  *Daniel Ferguson* | *2/19/2011* |  |  | *Project cost rework* |

**Abstract**

TIU tracking system is a capstone project conducted by Portland State University students and sponsored by Intel Corp. The system will be used to track locations of Test Interface Units (TIUs) in the Intel Validation Lab.

In this paper, we propose a general architecture for the tracking system. The main goal of the architecture aims to achieve a low-cost, yet effective tracking system. The infrastructure of the tracking system is built upon RF transceivers which communicate wirelessly in a mesh network. The asset tags will be attached to tracked devices (TIUs). These asset tags will broadcast messages to detectors. Messages that contain received signal strength indication (RSSI) values will be routed though the mesh network to a server. The server will analyze the received data, calculate the locations, and store them in the database. A web application will then be able to access the database and display location information in an interactive 2D map.

The primary figures of merit that were used in the decision making process include accuracy, power consumption, size, and cost. The last administrative idiom this paper conveys is a road map which will guide our team to the final objective. The means by which this is accomplished is through a tentative schedule, timeline, a list of milestones and expected deliverables.

Table of Contents

[1 Introduction 1](#_Toc285884025)

[2 Functional Specification 2](#_Toc285884026)

[2.1 Functional Requirements 2](#_Toc285884027)

[2.2 Non-Functional Requirements 2](#_Toc285884028)

[3 Design Proposal 3](#_Toc285884029)

[3.1 Topology 3](#_Toc285884030)

[3.2 Software 5](#_Toc285884031)

[3.2.1 Front-end Software................................................................................................ 5](#_Toc285884032)

[3.2.2 Back-end Software................................................................................................. 6](#_Toc285884033)

[3.3 Hardware 8](#_Toc285884034)

[3.3.1 Asset Tag.................................................................................................................8](#_Toc285884035)

[3.3.2 Detector................................................................................................................. 9](#_Toc285884036)

[3.3.3 Proxy…………………………………………………………………………………………………...….…....... 9](#_Toc285884037)

[3.3.4 Parts Consideration................................................................................................9](#_Toc285884038)

[4 Work Breakdown Structure 11](#_Toc285884039)

[5 Road Map 12](#_Toc285884040)

[6 Task Schedule 13](#_Toc285884041)

[7 Milestones 13](#_Toc285884042)

[8 Deliverables 14](#_Toc285884043)

[9 System Requirements 15](#_Toc285884044)

[9.1 Hardware Requirements 15](#_Toc285884045)

[9.2 Software Requirements 15](#_Toc285884046)

[10 Project Cost 15](#_Toc285884047)

[10.1 Microcontroller 15](#_Toc285884048)

[10.2 RF Transceiver 15](#_Toc285884049)

[10.3 Wi-Fi Transceiver 15](#_Toc285884050)

[10.4 Server Computer 15](#_Toc285884051)

[10.5 Total Cost Estimation 16](#_Toc285884052)

[11 Conclusion 16](#_Toc285884053)

[12 Appendices 16](#_Toc285884054)

# Introduction

Location aware computing has a bright future in the fields of personal navigation, personal security, prompt healthcare and entertainment. Furthermore, information on the physical location of asset tags can greatly help in urban search and rescue missions, as well as enable geographical routing in ad hoc multi-hop networks. The determination of physical location is sometimes referred to as location estimation, location identification, localization, positioning or geolocation identification.

Given the size and complexity of Intel validation lab, locating Test Interface Units (TIU) can be a tedious task as test equipment has to be broken down to identify the TIU it is using.

Chip manufacturers today face dramatic upheaval across the industry. Companies (such as Intel) need to find ways to remain profitable at a time when economic turbulence is curtailing consumer buying power. In the midst of these changes, plant managers must find a way to address the problem of improving processes while becoming more cost-effective. As part of this effort, Intel wants to automate tracking of the TIUs at different locations of the Intel lab, with the intention of wirelessly locating a TIU’s physical position at all times.

By tracking the location of the TIUs in the lab, Intel can minimize downtime and decrease labor costs caused by the size and complexity of Intel validation lab. This solution enables:

* Increased equipment utilization
* Decreased loss and theft
* Reduced costs
* Faster production
* Avoid time wasted on manual searches and inventory checks.

This proposal is organized as follows. In the next section, we present the project scope. In section 2, we present functional specification. In section 3, we present design proposal. In section 4, 5, 6, 7, 8, we present work breakdown structure, road map, task schedule, milestones and deliverables. In section 9, we present system requirements including hardware and software. In section 10, we present the project cost. Conclusions and appendices are presented in section 11 and 12.

This capstone project will be a prototype system that can wirelessly track the location of at least 10 TIUs in an indoor environment. The duration of the project will last 6 months.

# Functional Specification

## Functional Requirements

The key functionalities of the system are as follows:

* The primary user interface will be a web application
* Locations of TIUs will be displayed on an interactive 2D map within the user interface
* The user interface will show when an asset tag has low battery or when it is out of bounds

## Non-Functional Requirements

* The system will be able to track any TIU within a 10,000 sq ft area
* A tester in Intel lab occupies roughly 25 sq feet and the system will be able to locate the TIUs with an accuracy of 5 feet
* The locations of tracked TIUs will be updated within 10 minute intervals
* The asset tags must fit within a 1” x 1” x 1” dimension
* The asset tags will be permanently attached to the TIUs during operation
* The asset tags must use their own source of power. If the power source is a battery, then life must exceed one month
* The system will provide a user interface for adding and removing devices
* The system will provide a user interface to configure the geometry of the tracking area
* The locations in the database will be retained for 2 years
* The system will be able to operate properly in temperature range from -25°C to 85°C
* Parts and components will be available for manufacturing and maintaining
* All documentations regarding design, setup, use, and maintenance will be provided
* The system will not infringe upon existing patents, copyrights and trademarks and will comply with FCC rules and regulations

# Design Proposal

## Topology

Figure 3-1 shows a network of detectors that are geometrically located in a grid array. The density of the grid array will primarily be a ratio of accuracy and cost. For the scope of this project, the reasonable number of detectors would be from five to seven. The location of the proxy must be within range of at least one detector.

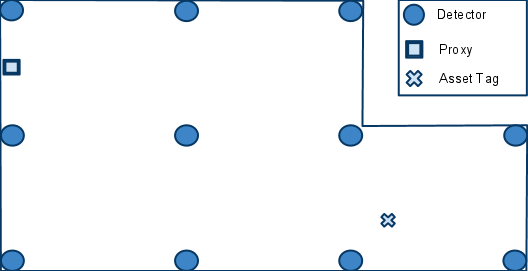


Figure ‑: A hypothetical room with detectors positioned in a grid array. Also shown is an arbitrarily placed proxy.

There are two distinct networks which compose this system: the front-end network, which includes all the physical nodes, and the back-end network, which includes all the logical nodes (Figure 3-2). The proxy is the gateway between the two networks.

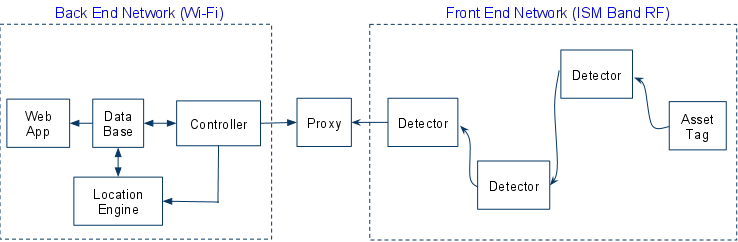


Figure ‑: Front-end and back-end network topologies.

There are three important aspects to the communications that take place in the front-end network.

* Each asset tag will broadcast a message periodically. Neighboring detectors will detect the broadcast, and determine the RSSI of the broadcasting asset tag.
* When a detector determines the RSSI of a broadcasting asset tag, it must then transmit this data towards the proxy using a routing table. The routing table will be configured in such a way as to ensure the data moves in a direction that is physically towards the proxy.
* The locationing application will periodically receive bursts of RSSI information from various detectors. Since it has no knowledge of how many detectors actually detected a broadcasting asset tag, it will wait a certain amount of time that assumes all the RSSI information from the various detectors has finished coming in.

Now we will further discuss some finer points of the second aspect. Figure 3-3 shows the paths taken as a result of an asset tag’s broadcasting. When an asset tag broadcasts, the nearest detectors are able to determine the corresponding RSSI values. The detectors that detected the asset tag broadcast are denoted by a connection with a green arrow. Each detector will have a routing table programmed into them. The routing table will contain information that steers the messages physically toward the proxy. In the example shown in Figure 3-3, the routes taken are in red. The blue dash paths are routes that exist but are not taken.

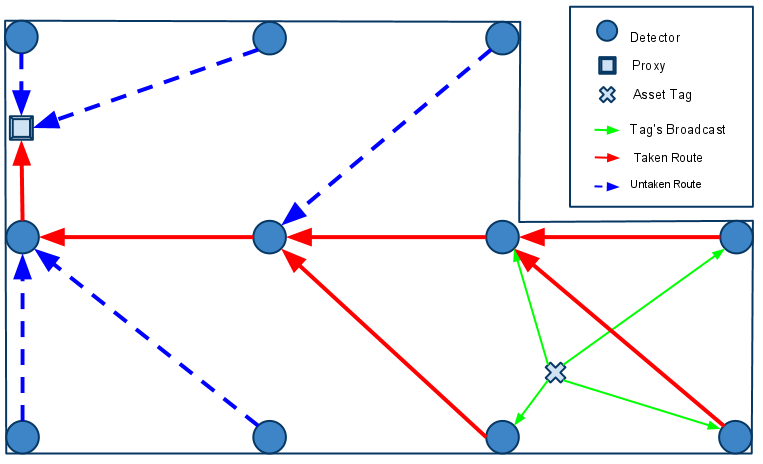


Figure ‑: An example of the routing paths as a result of an asset tag’s broadcasting.

## Software

### Front-end Software

The front-end software consists of the firmware for the detectors, and the asset tags.

For a detector to determine the RSSI of an asset tag, the asset tag must broadcast a message. The message has six fields, as shown in Figure 3-4. The way the fields are used depends on the origin of the message. There are two possible origins of a message. The first type of origin is an asset tag. A message can be identified as coming from an asset tag by examining the DID, SID, and RSSI value fields. If all three fields contain the value zero, then the message originates from an asset tag. Otherwise, the message originates from a detector.

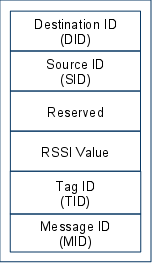


Figure ‑: Message format.

When receiving a message whose DID is not zero, a detector will do four things:

1. Replace the DID of the message with a DID contained in its routing table
2. Replace the SID of the message with its own ID
3. Replace the RSSI value with that inferred from the reception of the asset tag broadcast
4. Rebroadcast the message

An example of this message passing sequence is show in Figure 3-5.

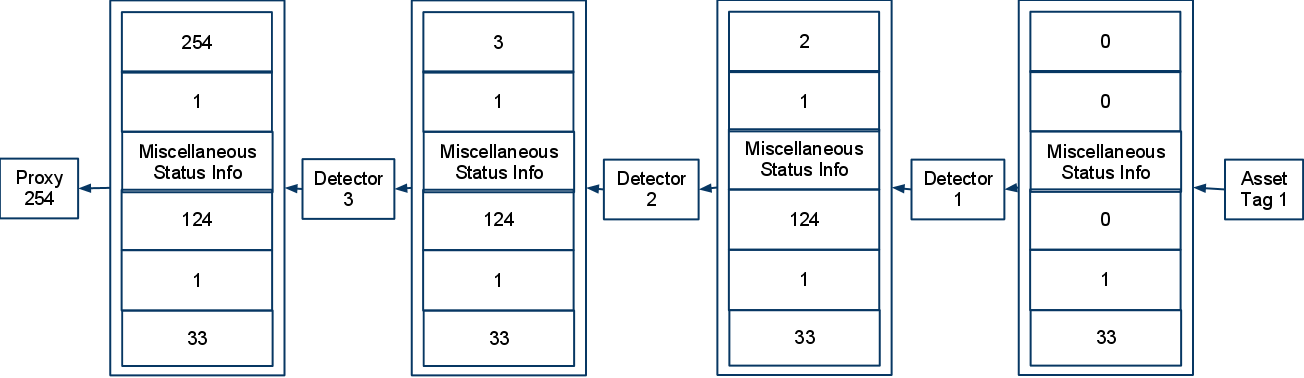


Figure ‑: A example of a message originating from a asset tag, and being relayed to a proxy.

### Back-end Software

The back-end network, composed exclusively of logical nodes, will rely on normal internet communication. This allows applications to leverage TCP/IP and many APIs that have been built upon it. Also, we gain the advantage of being able to have standard inter-nodal communication regardless of if they exist on the same computer or not. This increases the scalability of the system.

The back-end software system consists of four components:

* SQL Database
* Web Application
* Controller
* Location Engine

#### SQL Database

The SQL database is a central component that provides a link among the web application, the controller, and the location engine. Specifically, the database stores the following data:

* Location update interval
* Geometry of the tracking area
* RSSI-distance model or RF fingerprinting
* Identifications and locations of the detectors
* Identifications, locations, and battery levels of the TIUs
* Mapping table between asset tags’ ID and TIUs’ ID

We agree on using SQL since it is a standard that is easy to use, secure and scalable.

#### Web Application

The web application provides an interface for users to access the location information in the database. Its functionalities are:

* Display the locations and battery levels of all tracked TIUs on an interactive 2D map
* Allow users to search for a TIU via its ID. In case the ID is not complete or unique, the application will highlight the locations of all matched TIUs
* Allow users to view location history of the TIUs
* Alert users when a TIU is brought out of bounds or when it runs out of battery

The web application consists of two components: the client side and the server side. The client side will be implemented using modern web technologies such as HTML5, JavaScript, CSS3, and AJAX. The server side could be built upon one of the two most popular web server applications, i.e. Microsoft IIS and Apache.

#### Controller

The controller application is a GUI application that allows an administrator to manage the system. It allows the administrator to

* Modify the geometry of the tracking area
* Remap asset tags’ ID to TIUs’ ID
* Add or remove devices (detectors and asset tags)

The controller also receives RSSI data from the proxy, preprocesses them before feeding them into the location engine.

#### Location Engine

The location engine receives RSSI values from the controller and uses those values to calculate the locations of the asset tags. The derived locations will then be stored in the database.  
In case of using battery to power the asset tags, this application also receives battery level reports of the asset tags from the detectors and update the data in the database. This application should be written in a programming language that supports SQL database access and network communication.

The controller and the locationing applications are discussed herein as two separate applications. However, they can definitely be combined into one single application.

## Hardware

The infrastructure of the tracking system consists of detectors, asset tags, proxies and a server, as shown in Figure 3-6.

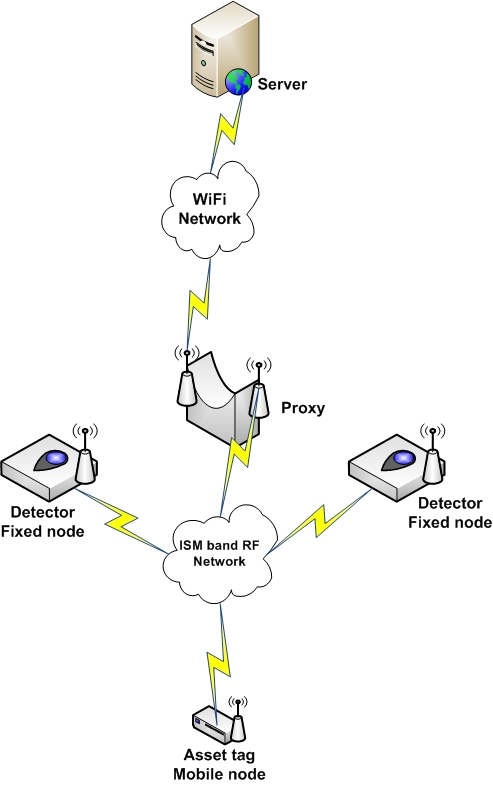


Figure ‑: An example topology for hardware infrastructure.

### Asset Tag

An asset tag consists of two major parts: a microcontroller, and a RF transceiver. The microcontroller controls the RF transceiver. The RF transceiver is responsible for communicating information to the system.

### Detector

A detector consists of two major parts: a microcontroller, and a RF transceiver. The microcontroller controls the RF transceiver. The RF transceiver is responsible for communicating information to the system.

### Proxy

The proxy consists of three major parts: a microcontroller, a Wi-Fi transceiver, and an ISM band transceiver. The microcontroller is responsible for relay data between the two transceivers.

### Parts Consideration

The parts below are our first but not final consideration of possible hardware combinations for this project. This would help us get a first perspective on the hardware solutions and estimate the possible cost of the system. These are selected carefully to make sure they meet these main requirements: small size, low power consumption and low price. Also, it is important that they are available and well-supported.

#### Microcontroller (MCU)

We consider two Atmel MCUs as the main processing units: ATmega and ATtiny.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Price (USD) | Size (mm) | Active current | Support | Sleep Current |
| ATmega | 4 | 9x9 | 9mA | Excellent | 1mA |
| ATtiny | 2 | 5x5 | 6mA | Poor | 1mA |

#### RF Transceiver

We consider three transceivers for main communication between the asset tags and the detectors: RFM12, RFM22, and XBee.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Price (USD) | Size (mm) | Active current | Support | RSSI | RF Band |
| RFM12 | 7 | 16x16 | 25mA | Excellent | Analog | 433MHz |
| RFM22 | 12 | 16x16 | 80mA | Poor | Digital | 433/868/915MHz |
| XBee | 23 | 24.38 x 27.61 | 50mA | Excellent | Digital | 2.4GHz |

#### Wi-Fi Transceiver

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Part | Source | Price (USD) | Size(mm) | Active Current | Support |
| WiFly Shield | SparkFun | 89.95 | 65x55 | Rx 40mA  Tx 210mA (Max) | Arduino Experimental Library |
| WiFly GSX | SparkFun | 49.95 | 21x38 | Rx 40mA  Tx 210mA (Max) | SMD Chip  Boasts EASY integration |
| MRF24WB0MA | Microchip | 26.57 | 31x18.4 | Rx 85mA  Tx 154mA | Requires using PIC microcontrollers to benefit from easy integration |

#### Battery

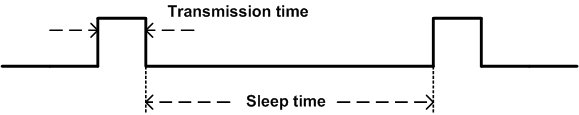
If the asset tags are powered by batteries, coin cell batteries seem to be a preferable choice due to the size of the tag. These are wide available in different capacities and prices

For example, Digi-Key has 3V-1000mAh, 0.96-inch-in-diameter coin batteries at about $3-$4.

Assuming that

* Transmission time: 100ms every 10 seconds
* Active current 100mA (RF transceiver + MCU)
* Negligible sleep current

Then, the approximate battery life is



This shows the battery meets our requirements.

This estimation is reasonable if the RF modules and the microcontrollers support µA sleep modes.

# Work Breakdown Structure

The Work Breakdown Structure (WBS) in Figure 4-1 provides a framework for the development of the TIU Tracking System using a hierarchical tree structure. The first two levels of the WBS (the root node and Level 2) define a set of 6 main tasks (1.0, 2.0, 3.0, 4.0, 5.0, and 6.0) that collectively and exclusively represent 100% of the project scope. At each subsequent level, the children of a parent node collectively and exclusively represent 100% of the scope of their parent node. Each children node should be done in order (Ex: 1.1, 1.2, 1.3, 1.4) to complete the main task (parent node 1.0).

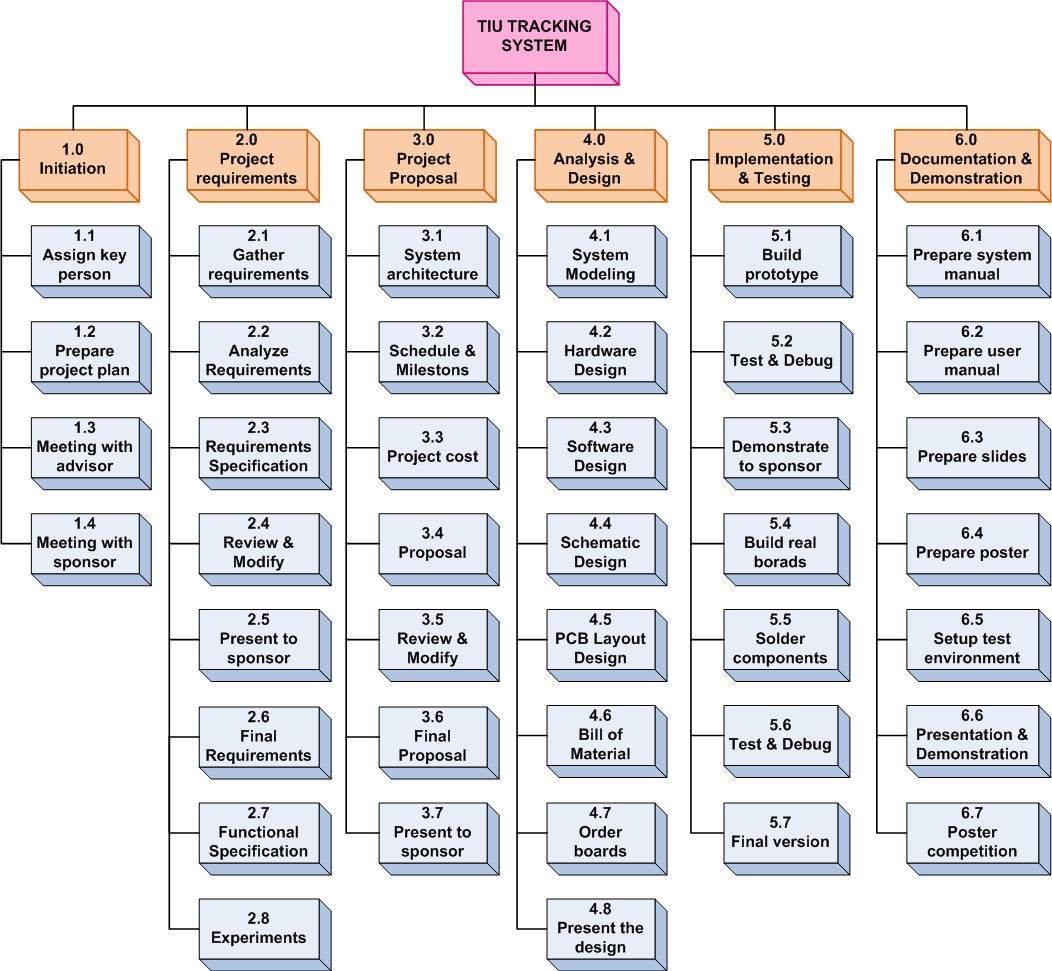


Figure ‑: Work breakdown structure.

# Road Map

The road map (Figure 5-1) shows main tasks that must be done in this project from January to June. The big arrow represents one month period, and the small arrow represents the dependency among tasks. Example: prototype task must depend on hardware design and software design tasks.

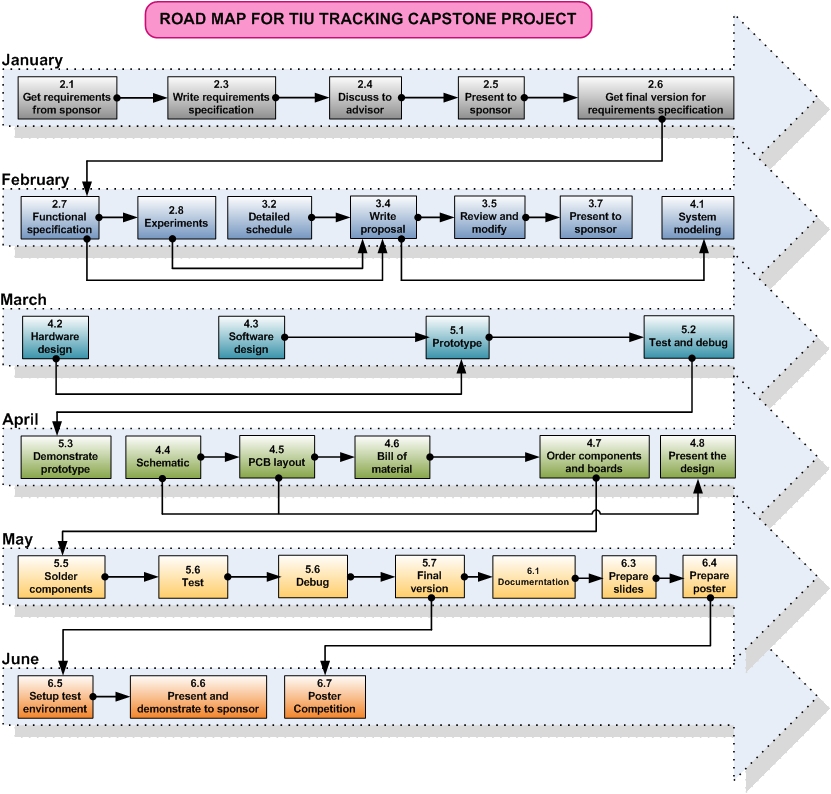
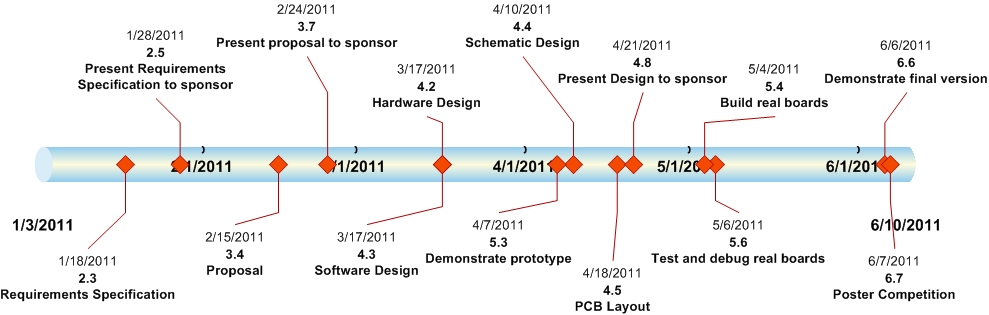


Figure ‑: Road map

# Task Schedule

Please see the Gantt chart that was attached to this document.

# Milestones



# Deliverables

|  |  |  |
| --- | --- | --- |
| No. | Deliverable | Tentative date |
| 1 | Requirements Specification | Jan 28, 2011 |
| 2 | Proposal | Feb 15, 2011 |
| 3 | Hardware prototype (breadboard)   * Three asset tags * Six detectors * One proxy | Apr 7, 2011 |
| 4 | Software prototype   * Web application * Controller * Location engine * Proxy firmware * Detector firmware * Asset tag firmware | Apr 7, 2011 |
| 5 | Hardware products (PCB)   * 10 asset tags * Six detectors * One proxy | Apr 18, 2011 |
| 6 | Software product (Release Candidate)   * Web application * Controller * Location engine * Proxy firmware * Detector firmware * Asset tag firmware | Apr 18, 2011 |
| 7 | Product Design Specifications   * Hardware Design * Software Design | Apr 21, 2011 |
| 8 | Documentations   * System maintenance * User guide | May 25 2011 |
| 9 | Final presentation and demonstration | Jun 6, 2011 |
| 10 | Poster competition | Jun 7, 2011 |

# System Requirements

## Hardware Requirements

|  |  |  |
| --- | --- | --- |
| No. | Device | Quantity |
| 1 | Asset Tag | 10 |
| 2 | Detector | 6 |
| 3 | Proxy | 1 |
| 4 | Server | 1 |

## Software Requirements

|  |  |  |
| --- | --- | --- |
| No. | Software | Quantity |
| 1 | Microsoft Windows operating system | 1 |
| 2 | Web server (IIS or Apache) | 1 |
| 3 | Development tools | N/A |

# Project Cost

## Microcontroller

There are many possible microcontrollers we could use to implement a solution. The ATmega 328 is a good choice because they are the Arduino technology platform. However, if under testing we discover power to be an issue, we have the possibility to move to a smaller ATtiny. The average price for any microcontroller we would consider will be between $2 and $5.

## RF Transceiver

The RF transceivers vary widely in capability and cost. More testing will need to be conducted before we fully understand the characteristics and implications of each transceiver. The most inexpensive transceiver runs for around $7, and the most expensive is around $20.

## Wi-Fi Transceiver

The Wi-Fi transceiver benefits from the fact that we need only one to complete the overall prototype. In general, a wisely configured system should not require many of these, relative to other hardware modules. The cost of the transceiver, therefore, has less impact on the overall cost of the system. In general, these devices run between $30 and $90.

## Server Computer

Included in the cost tabulation is the price for a server computer that executes all the back-end software. We do not expect numerical processing to be an extreme burden, and therefore do not require a high end computational platform. An entry level consumer PC can provide the necessary power required and costs on average $400.

## Total Cost Estimation

Table 10-1 shows the upper bound cost estimation of the tracking system.

Table ‑: Upper bound cost estimation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Manufactory | Description | Qty | Unit Price (USD) | Amount (USD) |
|  | **TOTAL** |  |  |  | **1255.00** |
| A | **HARDWARE** |  |  |  | **1255.00** |
| A1 | **Asset Tag** | |  |  | **450.00** |
| 1 |  | MCU | 10 | 5.00 | 50.00 |
| 2 |  | RF Transceiver | 10 | 20.00 | 200.00 |
| 3 |  | Other (battery, components, …) | 10 | 20.00 | 200.00 |
| A2 | **Detector** | |  |  | **270.00** |
| 1 |  | MCU | 6 | 5.00 | 30.00 |
| 2 |  | RF Transceiver | 6 | 20.00 | 120.00 |
| 3 |  | Other (battery, components, …) | 6 | 20.00 | 120.00 |
| A3 | **Server** | |  |  | **400.00** |
| 1 |  | Computer | 1 | 400.00 | 400.00 |
| A4 | **Proxy** | |  |  | **135.00** |
| 1 |  | MCU | 1 | 5.00 | 5.00 |
| 2 |  | RF Transceiver | 1 | 20.00 | 20.00 |
| 3 |  | Wi-Fi Transceiver | 1 | 90.00 | 90.00 |
| 4 |  | Other (battery, components, …) | 1 | 20.00 | 20.00 |
| B | **SOFTWARE** |  |  |  | **0.00** |
| 1 |  | Web Server | 1 | 0.00 | 0.00 |
| 2 |  | Development Tools | 1 | 0.00 | 0.00 |

# Conclusion

The tracking system consists of two primary parts: hardware and software. The hardware infrastructure is built upon RF technology. The number of detectors will be scaled in order to achieve the desired accuracy and efficiency. The locating algorithm is in software, and more experimenting will have to be conducted until we know for certain how we will implement it, but we are confident that we will converge upon a solution.

# Appendices

Include relevant information as needed to support the proposal. Appendices should include:

* Requirements specification document approval with original signatures
* Gantt chart