Stolen Object Tracker

Cycle 1 Report

By

Charles Baker

Denney Burkholder

Nathan Plotts

Leo Reyes

Steven Whaley

Submitted in partial fulfillment of the requirements for  
COMP 4710 Senior Design  
to the Department of Computer Science and Software Engineering,  
Samuel Ginn College of Engineering, Auburn University

Auburn, Alabama

March 4, 2014

Table of Contents

[2 Executive Summary (System Metaphor) 4](#_Toc381449277)

[3 Project Introduction 5](#_Toc381449278)

[3.1 Previous Development 5](#_Toc381449279)

[3.2 Intent This Cycle 6](#_Toc381449280)

[3.3 Future Work 6](#_Toc381449281)

[4 Requirements & User Stories 7](#_Toc381449282)

[4.1 Customer Requirements 7](#_Toc381449283)

[4.2 User Stories 7](#_Toc381449284)

[4.2.1 User Account Creation 7](#_Toc381449285)

[4.2.2 User Account Login through Web 7](#_Toc381449286)

[4.2.3 User New Device Registration 8](#_Toc381449287)

[4.2.4 Administrator Manages User Accounts 8](#_Toc381449288)

[4.2.5 User Views Device List 8](#_Toc381449289)

[4.2.6 User Views Device Data 9](#_Toc381449290)

[4.2.7 User Views Device Location on Map 9](#_Toc381449291)

[4.2.8 User Reports Stolen Device (Activates Device Tracking) 9](#_Toc381449292)

[4.2.9 User Installs Windows Service 10](#_Toc381449293)

[4.2.10 Laptop Automatically Connects to Open Wi-Fi 10](#_Toc381449294)

[4.2.11 Laptop Logs Keystrokes and Sends Data to Server 10](#_Toc381449295)

[4.2.12 Laptop Checks in with Server 11](#_Toc381449296)

[4.2.13 Geogram Automatically Detects when Stolen 11](#_Toc381449297)

[4.2.14 Geogram Sends Location Updates to Server 11](#_Toc381449298)

[5 Design Documentation 12](#_Toc381449299)

# Executive Summary (System Metaphor)

By: Leo Reyes

When an object is stolen, it often never makes it back to its owner. Although there are other tracking devices currently on the market, our product provides more data about the thief than a regular tracker can, and also provides a user-friendly online interface for viewing device related updates and managing user accounts.

There are two different tracking device options. The first is a laptop application that runs in the background, hidden from the user. When the laptop is stolen, the application can gather information from its environment and use this information to locate and retrieve the laptop, this information includes their IP addresses as well as a record of their keystrokes.

Another tracking option available is a GPS tracking device. This device is small enough that is can be placed inside most commonly stolen objects and will activate when notified by the owner.

A user-friendly online system, with a simple account registration and login, is available to device owners. When a device is stolen, the owner can easily activate the tracker by logging into the online system. The owner of the device can view location updates on a map through the online system, along with data related to specific devices including logged keystrokes and IP addresses. Support is also provided for registering and managing multiple devices.

# Project Introduction

By: Denney Burkholder

Over the past four weeks, our team has been working on creating a recovery program to track objects that are stolen without alerting the thief that he or she is being monitored.

We are designing for laptop computers and an object location device.  The goal for our project has been to create a product that will allow our customers to view their locations through our website, along with other information that may prove useful to recovering the device.  The need for a product such as this is undeniable when looking at the recovery rates for expensive devices (such as laptops, mobile phones, and tablets) once they have been lost/stolen.  Anyone who owns a desirable device can fall victim to thievery.  As such, they are our intended clientele.

By logging on to our user-friendly web application, the customer will be able to register their devices and access different information depending on the type of device.  For laptop computers, we have created a key logger that will be remotely activated by our server once the device has been reported missing on our website.  It runs in the background of the laptop, so the thief will not be aware that information is being gathered.  This file will be sent to our server whenever the laptop can get an Internet connection through available Wi-Fi networks.  The files can be viewed and downloaded from our web application. Our object location device is a self-aware tracking device.  It has GPS capabilities and an accelerometer allowing it to transfer its location along with the speed with which it is traversing to our server for viewing on our web application.  It communicates through cell towers, so a Wi-Fi connection is not necessary.

 Implementing our project involves many different components.  We have designed and coded a central server, database, applications for gathering data on laptops and web pages for communicating between the stolen devices and customers, and storing relevant information for recovery.

## Previous Development

By: Charles Baker

Previous development was performed in the architectural spike phase. Development was focused on the server and Windows service. Goals for the architectural spike were to get the most important features to a point where it was evident we could continue development without any major problems that may cause us to miss our final deadline.

Goals for the server were to get most planned functionality working in a basic state. The server was designed in 2 primary parts, the application server and web server. Goals for the web server included: embedding Google Maps into a page, setting markers on the map, and connecting to the application server. Goals for the application server included: communicating with the Windows service, communicating with the database, communicating with the web server, and moving data between different connections on request. The goal to connect the web and application server was not met because of difficulties with GlassFish. Furthermore, we were not able to work on sending requests do to time constraints and because most involve the web server. Otherwise, we made satisfactory progress on the remaining goals. We created the html pages and were able to embed Google maps into it. We were also able to add markers to the map by sending coordinates through a web socket. The application server was able to communicate to the Windows service by a TCP connection. We developed a messaging protocol that both server and client (service) adhere to when communicating. The server was also successfully communicating to the database by the end of the architectural spike. We were able to read and write to a database containing all values we predicted we would need in the future.

On the Windows service, by the end of the architectural spike we wanted to have it start automatically, communicate with the server automatically, identify itself, and log keystrokes. By default, when the service is installed it launches on startup. The service was able to identify itself by using a MAC address of a network adapter. This method mostly worked, but a virtual adapter appearing could cause problems in selecting the same MAC address each time. Connecting to the server was achieved by creating a thread to handle the connection. If connection was ever lost, it would simply try to reconnect. While connected, the service could respond to different commands sent by the server. Commands implemented where: start key logger, stop key logger, return key logs, and send an IP trace. Key logging was achieved, but was in a very basic state at the end of the architectural spike. It was able to record the key being pressed, but did not consider keys that were being held down. This means that it did not record the case of letters or other effects of key combinations being pressed.

## Intent This Cycle

By: Charles Baker

We decided it would be beneficial to port the existing server to the Google Go language. This allowed us to have a single server, instead of both an application server and web server. This is possible because Go allows application code that contains HTTP handlers. It also provided better support for multithreaded applications, which is important for our server that must manage connections to many devices and web connections. We believed that In the long run, the change to Go would speed up the development of the server by requiring a fewer number of components that must interact.

Another one of our primary goals this cycle was to incorporate the Geogram into the system. The first step was to send messages between the server and Geogram. We decided the communication would be done through an Android phone. An Android application would be responsible for connecting to the server via TCP and interfacing between TCP and SMS communication. The server must be able to send the appropriate commands and interpret received messages.

Work on the Windows service was also planned for this cycle. We planned to have the service check in to the server in set intervals. If the server notified the service that the laptop was stolen, it would constantly try to reconnect if the connection is ever lost. We also planned to improve the key logger by capturing when a key is modified by shift or capslock. We also planned to capture key combinations when multiple modifier keys are held down.

## Future Work

By: Charles Baker

A large portion of the remaining work is derived from finishing the website that will allow user to interact with the system. The registering and login system requires more work so that users will be restricted from certain pages if they are not logged in. We must also associate devices with the users that register them and be able to display relevant information on the website. The users must also be able to send commands to the server from the website, such as manually reporting a device stolen.

On the server we need to improve how devices are handled. One thing this includes is setting how automated commands will be sent to devices. Information must be gathered from registered devices in regular intervals to detect the possibility of being stolen. For example, Geogram devices must be sent commands in a way that maximizes battery life but provides up-to-date information when once are stolen. Devices must be read from the database to memory in an efficient way and removed from memory when no longer needed. One occasion where this is important is when a user logs in and wants to view his/her devices.

# Requirements & User Stories

By: Charles Baker

## Customer Requirements

Our customer described an open ended project that resulted in being able to track stolen objects. He gave us a few requirements that we must adhere to. First the device must be able to locate itself. Secondly, the device must identify itself. Thirdly, the object must be able to communicate this information to the user. He also mentioned, when applicable, we should consider the battery life of any solutions we implement.

We were also given examples of possible implementations. One example was a device that could be hidden in large shipments and tracked if anything were to happen. Another was a device that is disguised as an expensive object that a thief would likely grab during a robbery. Additionally, the device could be an existing electronic device, such as a laptop that sends its IP address as a means of location.

We were to present our ideas to the customer and he would decide which solutions he preferred we implement. One solution he wanted us to implement was an application for laptops, that could send its IP address for the purpose of tracking. Additionally, he wanted us to integrate the existing Geogram device into our system and use its GPS location to track it.

## User Stories

By: Charles Baker

### User Account Creation

Summary: The user enters account credentials into the website and a new account is registered and stored in the database.

Description: The user enters his/her first name, last name, email address, phone number, password, and confirm password. The user presses the register button. The server ensures all fields are valid, and if so creates a new user in the database with the entered information. If any fields are invalid, the user is notified which field(s) caused registration to fail.

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### User Account Login through Web

Summary: A user with an existing account logs in.

Description: The user accesses the website and enters his/her email address and password. The server verifies the credentials and logs the user in if they are correct. The logged in user is redirected to his/her main page. If the login fails the user is notified that login failed.

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### User New Device Registration

Summary: The user registers a new device on the website.

Description: The user goes to his/her device page and selects to register a new device. The user adds relevant device identifiers. The server checks that the information is valid, and if so associates a new device with the user in the database.

.Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### Administrator Manages User Accounts

Summary: The administrator views user accounts and can modify them.

Description: The administrator views a full list of user accounts. The admin can select an account to view its information. The administrator can modify account information or terminate the account.

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### User Views Device List

Summary: The user views his/her devices.

Description: The user goes to a page displaying all devices registered by him/her. The devices are displayed in a list.

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### User Views Device Data

Summary: The user views information about a registered device.

Description: The user selects a device on on his/her device page. Information and commands relevant to the device appear. This can include (depending on device) name, ID, device type, option to load other data (key logs, etc.), and the view the last location (IP or map depending on device).

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### User Views Device Location on Map

Summary: The user selects to view information of a GPS device and a map is displayed on the page with location data for the device.

Description: The user selects to view information for a GPS device. Google Maps is embedded in the page. The maps contains a marker for the most recently recorded location of the device. If the user selects to view previous locations of the device, the location data will appear as markers on the map colored differently than the current location.

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### User Reports Stolen Device (Activates Device Tracking)

Summary: On the device list, the user selects a device and reports it stolen.

Description: The user is viewing a certain device. The user chooses to mark the device as stolen. The server attempts to keep gathering up-to-date information about the device.

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### User Installs Windows Service

Summary: User installs the Windows Service to their laptop.

Description: The user runs the installer for the Windows service. The installer places the executables and resource for the service and kelogger so that they will run on startup.

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### Laptop Automatically Connects to Open Wi-Fi

Summary: Laptop tries to connect to open or known Wi-Fi if stolen.

Description: When the laptop is marked as stolen it will try to maintain a connection to the server. If the laptop Is unable to connect because it has no Internet connection, it will attempt to connect to open and known Wi-Fi so it can communicate with the server.

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### Laptop Logs Keystrokes and Sends Data to Server

Summary: When stolen the laptop logs keys and sends them to the server.

Description: If the laptop is flagged as stolen, it turns on the key logger. The key logger stores keystrokes in a logs files. Periodically the service sends the contents of the log files to the server for viewing.

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### Laptop Checks in with Server

Summary: The laptop periodically connects to the server when not flagged as stolen.

Description: The laptop will connect to the server periodically when it is not flagged as stolen on the laptop. If the device is flagged as not stolen on the server, the server will tell the laptop it is not stolen and the laptop will wait again before checking in. If the server has the laptop marked as stolen, it will tell the laptop that it is stolen and it will try to maintain communications.

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### Geogram Automatically Detects when Stolen

Summary: The Geogram detects if it is stolen based on movement and location data.

Description: The accelerometer options are set so that the Geogram sleeps until it detects movement. Once it wakes up, location data will be gathered do determine if it is stolen. If the server determines there is potential for it to be stolen the user will be alerted.

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

### Geogram Sends Location Updates to Server

Summary: The server sends a request for th Geogram’s location due to the user marking it as stolen or the Geogram detects movement.

Description: The user marks the Geogram as stolen or it detects movement and sends the server a message. The server periodically requests the location of the Geogram.

Hours: Total Planned:   
Planned this cycle:   
Total Actual:   
Actual this cycle:

Coder:

Tester:

Reviewer:

Status:

# Design Documentation

3-tier architecture – Server, database, and client (Web)

## Server components (Leo Reyes)

### Central Server

The central server is the entry point of the server. It initializes and starts the modules of the server, and initializes the communication channels used by the modules.

The central server connects all of the server modules together. All components send and received Requests through the central server. When a request is received, the central server redirects the request to the proper module.

### RequestProtocol

The modules of the server communicate with a requesting protocol. Using a requesting protocol allows the server to be organized. It also allows for adding and removing server modules with less effort, because it allows each module to have a high cohesion.

The protocol contains opcodes for modules to use when creating requests.

A requests contains an id, the destination of the request, the source of the request, an opcode, a payload, and a response channel that the response to the request will be sent over.

### Web Server

Http handler – handles http requests received and sends back an http response to the request. The response will contain all of the files needed for the client view. A typical response would contain files such as images, style sheets, scripts, and the html file.

Websocket handler – handles websocket requests made to the server. When a websocket request is received, the handler creates a new connection to the client and registers the connection in the Web hub.

WebClientConnection – Middleman between websocket and the web hub. Reads in messages from the websocket and pass it on to the hub. Also receives messages from the hub and passes it to the websocket.

Web hub – Maintains a list of connected clients as WebClientConnections; creates a channel for communication to and from those clients

Client Web Sessions – when a web client makes an http request to the server, a cookie-based session is created. A copy of the session is stored in the server. When a client logs in, the contents of the cookie are changed and represent a session for the client. The sessions are used for access control and to present the clients with relevant data.

RequestProtocol handler – handles incoming and outgoing requests made from the other server components.