|  |
| --- |
| Northern Arizona University |
| Final Report |
| CS 486c - Localization |

|  |
| --- |
| Blayne Kennedy, Kimberly Oyama, Daren Rodhouse, and Chihiro Sasaki  4/26/2013 |

Contents

[Problem and Solution Statement 2](#_Toc354667070)

[Part 1: Problem Statement 2](#_Toc354667071)

[Part 2: Problem Solution 2](#_Toc354667072)

[System Improvements 3](#_Toc354667073)

[System Impact 3](#_Toc354667074)

[Process Overview 4](#_Toc354667075)

[Team Roles 4](#_Toc354667076)

[Project Management Framework 5](#_Toc354667077)

[Design methodology 5](#_Toc354667078)

[Deliverables 5](#_Toc354667079)

[Requirements 5](#_Toc354667080)

[Solution Statement 6](#_Toc354667081)

[Overall Solution 6](#_Toc354667082)

[Functional Specifications 6](#_Toc354667083)

[Architecture Overview 7](#_Toc354667084)

[As-built Design (all together) 9](#_Toc354667085)

[Usability Testing and Future work 15](#_Toc354667086)

[Conclusion 15](#_Toc354667087)

# Problem and Solution Statement

MoneyClip Mobile (MCM) is a web-based payments infrastructure providing fee-free transactions to customers and merchants through the use of mobile smartphones. Mr. Joshua Cross of Hermes Commerce, Inc. is planning to make this mobile payment system more robust by providing targeted advertisements, coupons, and managing customer’s loyalty programs.

This capstone project will develop the localization features for the MCM advertisement, coupons, and loyalty network. The software will be developed for the existing MCM network and database to ease the transition of integrating this feature. The team will be developing the same copy of the existing network and database on a local server to develop in and test the added feature. The prototype of the localization features implemented on the database, web service, and iOS/Android applications will be presented to Mr. Cross at the end of this project.

## Part 1: Problem Statement

This project is sponsored by Mr. Joshua Cross, CTO of Hermes Commerce Inc. in Flagstaff, Arizona. Hermes Commerce Inc. produces an application called MoneyClip Mobile (MCM) that allows users to transfer funds to and from other users, merchants or customers, without involving physical money or credit and debit cards. Mr. Cross is responsible for the development of MCM as well as for the business aspects of the product. MCM addresses the need of users to exchange funds regardless of distance.

The general problem facing MCM is competition with popular technologies such as Google Wallet and Square. Mr. Cross’ goal for overcoming this challenge is to combine all the functionalities of the competition into one application and find a niche for that application. In order to accomplish this, MCM application needs to be able to transfer funds between customers and merchants. In addition, the application needs to determine when a customer is making a purchase from a merchant in order for the merchant to push the charge to the customer’s MCM account. Also, MCM will give customers control over the frequency and types of advertisements or coupons they receive, and it will give merchants options for when to push advertisements or coupons depending on customer trends.

This project involves implementing a localization function that notifies merchants when MCM customers are in their vicinity, allowing the merchant to push a payment request to the customer’s MCM account if the customer makes a purchase. This functionality allows true moneyless transactions. The third and final chunk of functionality that we will be developing is the data analysis algorithms that will be used for effective direct advertising. We will be implementing trend finding algorithms that will be used for direct advertising. These algorithms will use past user transactions to find patterns and compute probabilities that will be available to merchants to aid their advertising decisions.

## 

## Part 2: Problem Solution

### System Improvements

One piece of functionality we are adding is the ability to allow customers to select the amounts and sources of advertisements that are pushed to their account via the iPhone application, Android application, or website. They should be able to customize their settings to choose differing levels of participation from the advertisements. Merchant settings should also be available for change on the website or either mobile application.

The next function being added is the analysis of customer traffic to find patterns that merchants can use for direct advertising. The merchants will have the option to push an advertisement depending on the chance that a customer will respond. For instance, if a customer just ate at a restaurant downtown, there is 40% chance that they will get drinks when they are done. This information will be available to merchants so a bar can set their account to push an advertisement whenever a customer makes a purchase at a restaurant downtown. In this situation, there is a very low chance that the customer will eat again, so another restaurant will probably not push an advertisement to this customer. Related to this, advertisements must be pushed to customers with minimal time lag to make sure that they have not left the area before they receive the advertisement.

Of the data stored by MCM, we will use the customer transaction history to determine purchasing patterns. The algorithms we implement will produce statistics for merchants. The pattern recognition algorithms will analyze past customer transactions in order to predict future purchases. Merchants will be able to look at the probability of a certain customer making a purchase from them and determine whether or not they want to push an advertisement to that customer. The localization functionality requires additional user settings, such as types of advertisements or coupons they are accepting on their account.

### System Impact

Our additions will make MCM a more comprehensive and robust commerce application. Therefore, our sponsor will have a better product to pitch, potentially making his job easier. The development team of MCM will be able to utilize localization activity code from the Android and iOS applications and use them to further their development of the applications.

By using MCM with localization features, there will be fewer steps in the money exchange, and thus transactions will be faster and lines will move faster. Theft may also decrease because there would be less cash to steal, making it not worth the effort and consequences.

With the added localization functionality, MCM becomes a comprehensive commerce application. Many other applications focus on certain aspects of commerce but none so far combine them. MCM will have the potential of removing the need for physical money entirely because you will be able to move money between customers, and between customers and merchants regardless of distance. Merchants will also be able to advertise more effectively, and customers will be able to tailor their account for advertisements and coupons that they are interested in.

# Process Overview

## Team Roles

Team Leader: Kimberly Oyama

The team leader will be the main contact with between the team and Dr. Georgas, the team mentor. This person will keep track of the project tasks (who they are assigned to, when they are due, and their progress) and maintain communication lines between team members in case they aren’t keeping each other in the loop. This person will handle any conflicts that arise throughout the course of the project, with the exception of version conflicts which are the responsibility of the two programmers involved.

Customer Communicator: Kimberly Oyama

The customer communicator is in charge of making sure the customer is receiving the information they need about the project in a professional manner. This person will relay any information they receive from the customer to the rest of the team (the team leader will forward pertinent information to Dr. Georgas). When the team has information or questions for the client, it is the communicator’s job to format the message according to the client’s preference and clearly express the team’s intentions.

Recorder: Daren Rodhouse

The recorder will take notes during the meetings. This job may fall on more than one person because it is difficult to engage in conversation and take notes at the same time. This will be done by a pair of members so that each can be involved and keep track of what the other contributes, such as main recorder and assistant. This pair can change for each meeting. One of the pair will be assigned the agenda for the next week’s meeting and the other will be responsible for compiling the minutes and sharing them with the other team members, which includes adding them to the repository and sending a notification email.

Architect: Blayne Kennedy

The architect will make sure that our requirements document matches the client’s intended system. He will also make sure that, when we start implementing the system, it matches our architectural decisions made in the design phase.

Release Manager: Chihiro Sasaki

The release manager will make sure that we are submitting the correct versions as deliverables in class and to our client. This person will keep track of the control logs so we will know what builds are clean. It is also the release manager’s responsibility to review commit logs for consistency and accuracy. In the case that a release or commit breaks the system, it is the responsibility of the release manager to identify the problem and work with the responsible programmer to fix the issue as soon as it arises.

## Project Management Framework

We reported to our mentor on a weekly basis. During the meetings we would first hand in our weekly progress report and go over the tasks we were currently working on. Also during these meetings we would get any feedback on current projects and deliverables.

For managing and monitoring progress we used a program called YouTrack that we would assign issues and items that needed to be done to team members and once it was fix or done we would update YouTrack. As for monitoring the progress of the project YouTrack has statistics on what has been done and still needs to be done as well as who has done what and in what time frame. We would go over the data about every month to make sure everyone was making progress to the overall goal.

The decision process was where if there ever was an issue we would discuss the issue and both parties would give their input on it. After that we would then vote on how to resolve the issue and the team leader would decide on ties.

## Design methodology

We used Scrum to develop the project due to the nature of capstone projects as well as being a good fit for mobile applications. This allowed us to rapidly produce a first prototype and then modify it based on input both from our three groups of user testers as well as from our testing.

# Deliverables

## Requirements

The requirements acquisition process started with the group brainstorming with the project description to come up with the initial project understanding and requirements. The next step was to meet with our client to get a better idea of what he wanted out of the project. We then went through a cycle of refining, modifying, and confirming the requirements until our client and our faculty mentor were satisfied with the expected project outcomes.

The main functional requirements are that the deals must be meaningful in terms of location, the merchant and client must be able to interact when they are within a certain distance of each other, and the apps must have the basic mobile commerce functionality such as requesting or accepting payments. The main nonfunctional requirements are that the system must work as a website and, on Android and iPhone devices, the apps need to be user friendly for overall success, the localization features need to be accurate, and the data we transfer and store needs to be minimal for scalability. The environmental constraints we had were the operating system fragmentation on the Android, we had to use java and xml for the Android, we had to use objective-c for the iPhone, we were bound to use the Google Maps API for the localization and map features because we chose Google Maps for its popularity, and we needed to make sure that the direct deals arrived with the user in a timely manner.

In the Requirements Document, the Functional Requirements section includes the essential requirements are listed for the iOS/Android applications and the web service. Each of the requirements for different pieces of the system is further divided down into users and merchants, the two main stakeholders of the system. Next, the Environments Requirements section outlines some of the constraints and environmental requirements to be considered in the system design. This includes limitations from the existing infrastructure, considerations about the content filtering system for advertisement creators, Google API limitations for using Google Maps, language considerations, and device operating system platform considerations. The Non-Functional Requirements section introduces the major non-functional requirement considerations to the system. This includes accessibility, expandability, maintainability, and usability concerns that are important to the usefulness and further development of the MCM system. Each of the non-functional requirements is accompanied by a description, the condition in which they are valid, and a threshold of verifiability. In the Potential Risks Section, the important risks pertaining to the localization aspect of MCM are covered. They include any potential risks that affect any of the stakeholders involved in a relevant manner, along with their corresponding impact and possible mitigations. The risks include security of localization integration, privacy protection interference to geolocation, the potential of merchants pushing too many advertisements and coupons to the point of annoyance, loss of cell service, inaccurate geolocation, failure of the MCM server, and the potential of user mobile devices recognizing more than one merchant in its proximity.

## Solution Statement

### Overall Solution

### Functional Specifications

Our system required two user interfaces, one for merchants and one for customers, developed on both a website and mobile device. The main functionality for these systems includes requesting and accepting payments, changing any user settings, and viewing a map that shows nearby merchants. To create this system it was essential that we allowed the merchant and customer applications to communicate with each other so that transactions could happen between the two. The system also required an advertising/deal feature. The deals are sent to customers and can be based on criteria such as their age, gender, or shopping patterns. The deals also are sent to customers based on their location and, therefore, had to be sent out in a timely fashion so that the deal would still be relevant to the customer’s location when they received it.

### 

### Architecture Overview

Overall System Architecture

The entire system consists of two types of user interactions: customer and merchant. Each interaction is supported by three different platforms, iOS 5 running on an iOS device, Jelly Bean 4.2 running on an Android device, and a website. Each of the six interfaces, as seen in Figure 1, connects to web services on the server, which then interact with the database. The interfaces do not directly interact with each other; the web services acts as an intermediary among different interfaces.

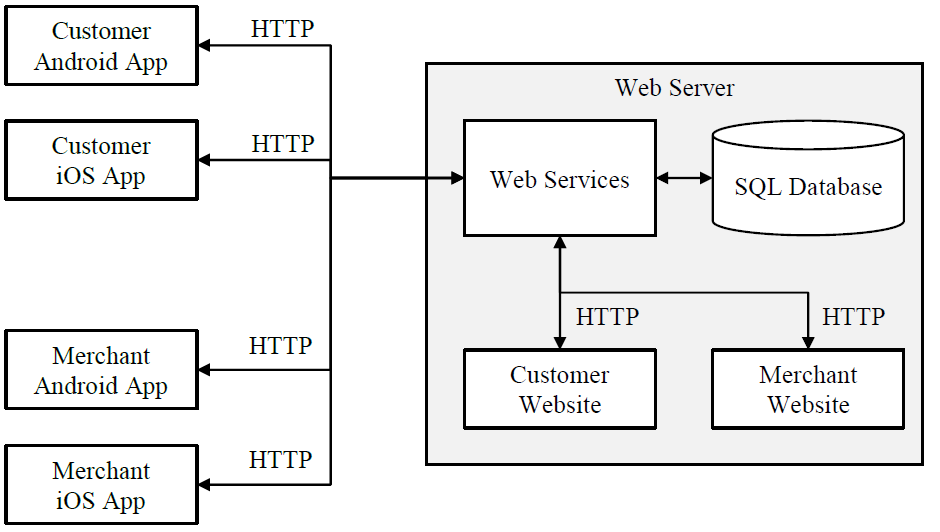


Figure 1: Overall system with its submodules and their interactions.

The four mobile device interfaces are placed external to the web server. The two website interfaces interact internally within the web server since the web services and the website live on the same server.

As seen in Figure 1, the main data flow between mobile devices and the web service happens through an HTTP request, where data passed is packaged in a JSON string. The customer and merchant website components also interact with the web service through an HTTP request. The web services interact with these interfaces in a RESTful way, using only POST, GET, PUT, and DELETE operations.

The server(s) host the web services, database, and the websites, seen in Figure 6. The web services take requests from the user interfaces and query the database, performing the requested update or returning the requested information. The web service uses a RESTful architecture, which only uses the basic operations of POST, GET, PUT, DELETE. Information is passed from the user interfaces through an HTTP request, where parameters are passed as JSON strings.

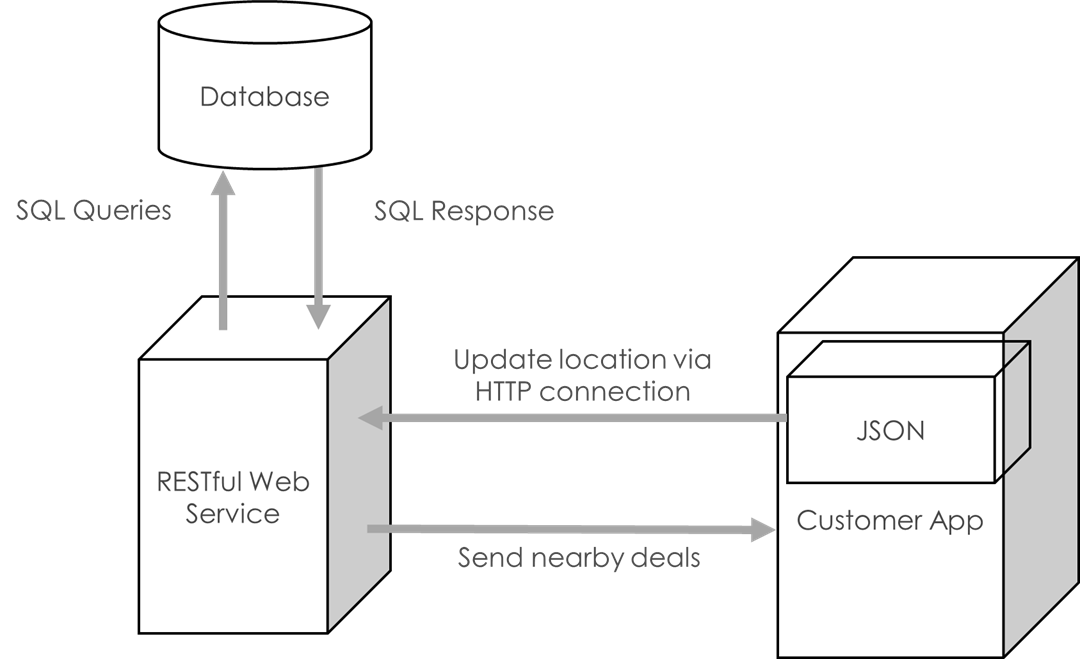


Figure #: Architecture for the localization functionality for direct deals

The customer interfaces allow the customers to interact with the MoneyClip Mobile web service. The main features include viewing account details, changing account settings, accepting charges from a merchant, changing the localization settings, viewing nearby merchants, and receiving notifications of new advertisements and coupons.

The merchant interfaces allow merchants interact with the MCM web service. The main features include viewing nearby customers, charging customers within their vicinity, and viewing recent transactions.

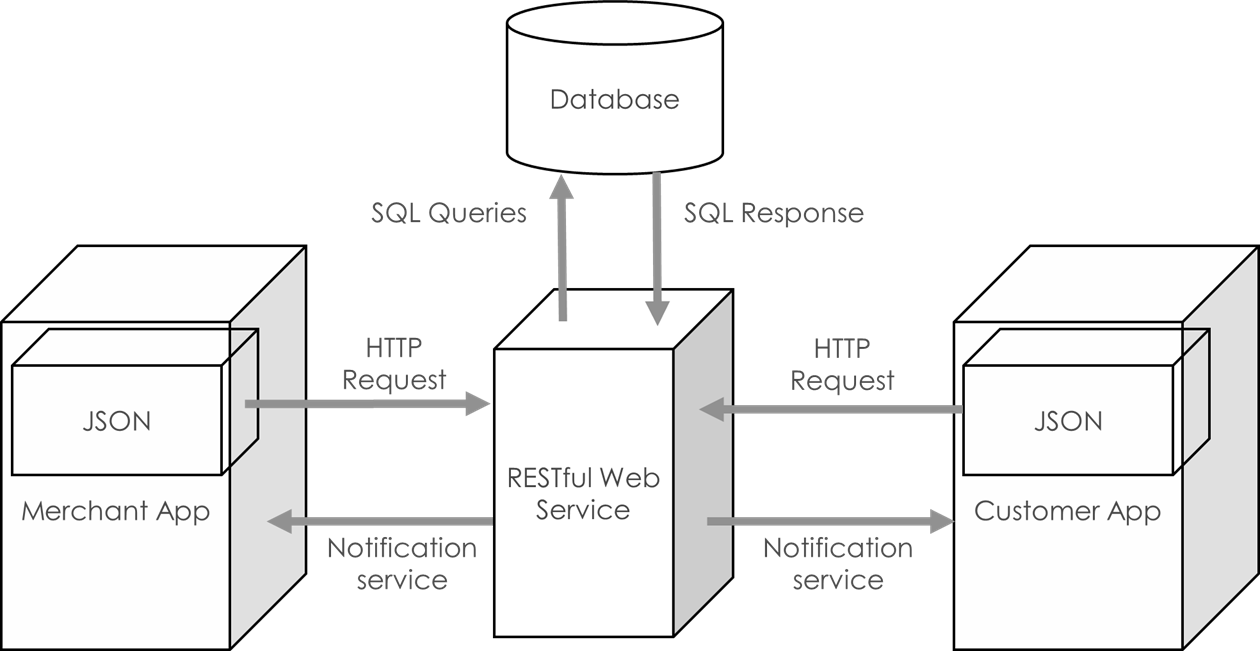


Figure #: Architecture for the localization functionality between the customer and merchant.

### As-built Design (all together)

Customer Interfaces

The customer interfaces allow the customers to interact with the MoneyClip Mobile web service. The main features include viewing account details, changing account settings, accepting charges from a merchant, changing the localization settings, viewing nearby merchants, and receiving notifications of new advertisements and coupons.

Website

The customer interface for the website provides the basic functionality for the generalized customer interface, as seen in Figure 2. The difference between the mobile applications is its lack of updating the customer’s current location using a network provider.

Ad/Coupon Settings

Home

Account Settings

Merchant Maps

View Charges

Recent Transactions

Login

Logout

Create Account

Forgot Account

Figure : Customer website module with its submodules and their interactions.

Mobile Application

The customer application module consists of several submodules shown in Figure 3. When the app is first opened, the initial login page allows registered users to access the core submodules of the system. The login page also allows new users to create an account or for registered users to retrieve their account information.

When a notification is pushed to the device about nearby deals, clicking on the deal opens up the MCM app. This will allow the user to take advantage of nearby deals without having to click on multiple pages to get to the deal.

Create Account

Login

Home

Settings

Merchant Map

Search

Localization Settings

Awaiting Transactions

Ads/ Coupons

Transaction History

Update Location

List of Merchants

Specific Merchant

Logout

Forgot Account

*Figure 3: Customer app module with its submodules and their interactions.*

Merchant Interfaces

The merchant interfaces allow merchants interact with the MCM web service. The main features include viewing nearby customers, charging customers within their vicinity, and viewing recent transactions.

Website

The website consists of all features of the merchant mobile application, in addition to several functionalities that are unique to the website. The merchant website architecture can be seen in Figure 4. The website has two pieces of functionality that the mobile applications do not allow: creating ads and creating coupons. These features are only implemented on the website because of the complexity of the forms; they would not translate to the screen size of mobile devices well.

Create Coupon

Home

Account Settings

Create Ad

View Charges

Recent Transactions

Login

Logout

Create Account

View Nearby Customers

Forgot Account

Figure : Merchant website module with its submodules and their interactions.

Mobile Application

The mobile application consists of the features available to the website, minus the feature to create ads and coupons. The mobile app architecture can be seen in Figure 5. The mobile application pulls a list of available ads and coupons from the web service, which they can then push to customers from their app. The ads and coupons are pre-approved before they are posted on the list, which addresses one of the risks of inappropriate ad content.

Create Account

Login

Home

Settings

Charge Customer

Awaiting Transactions

View current Ads/Coupons

Transaction History

Nearby Customers

Logout

Forgot Account

Figure : Merchant app module with its submodules and their interactions.

Server

The server(s) host the web services, database, and the websites, seen in Figure 6. The web services take requests from the user interfaces and query the database, performing the requested update or returning the requested information. The web service uses a RESTful architecture, which only uses the basic operations of POST, GET, PUT, DELETE. Information is passed from the user interfaces through an HTTP request, where parameters are passed as JSON strings.

*Figure 6: Server module with its submodules and their interactions.*

Server

Web Services

Mobile User Interfaces

Web User Interfaces

Database

# Usability Testing and Future work

We conducted three stages of usability testing. The first included two participants who separately went through scenarios on both applications. This stage helped us iron out the testing process and helped us fix some navigation issues before we moved to the next stage of testing.

The second stage of testing involved three pairs of participants. Each pair went through scenarios together on both applications. They gave us their feedback through discussion and the survey that we planned on using for the final stage of testing. Minor errors were found in the second stage and they were fixed before we proceeded to the final stage of testing.

The final stage of testing included 30 participants -15 pairs of testers. For this stage, we set up a camera to film button presses on the mobile devices. We also hooked the participants up with microphones so that we could record their conversation to try to help understand their thought processes. The main issues that we found during the final stage of testing were: Users expected map icons to be clickable and have some functionality, application navigation needed improvement, and some screens did not display meaningful feedback to help the users understand what was going on. After these issues were all fixed, the applications seem to be highly usable.

# 

# Conclusion

In the MoneyClip Mobile Localization project, the requested localization features were built and their corresponding documentation were created for our client, Dr. Joshua Cross. The localization features consisted of the main two parts: interaction between the merchant and customer within proximity and targeted deals for the customer. These localization features were implemented in the customer and merchant applications for the Android and iPhone.

The localization features were implemented using Google Maps and the devices’ built-in location providers. The accuracy of localization using GPS and Network providers, which includes Wi-Fi access points and cell towers, have been researched and implemented. The method in which these localization features were implemented, as well as research information on statistics in terms of accuracy, have been documented for our client.

Because our client wanted a proof-of-concept type of implementation for the localization features to add to the existing MoneyClip Mobile infrastructure, documentation was highly stressed throughout our project. Thus, we are providing Dr. Cross with several types of documentation to aid in his development, which includes the design specification document with the higher-level architecture and lower-level details of the entire API, a copy of our Doxygen-created API, documentation regarding Google Maps and location features of the phone, and a comprehensive document specifying developer notes and erratas that will be helpful for our client. All of these documentation, as well as the actual code, will be packaged onto a single USB drive and handed over to Dr. Cross.

Dr. Cross has been satisfied with the team’s progress throughout the project, and he is excited to receive the corresponding documentation for our localization features. The team is also satisfied overall with what we are providing to our client at the end.