# GO_SYNC - A FRAMEWORK TO SYNCHRONIZE CROWD-SOURCED MAPPING CONTRIBUTIONS FROM ONLINE COMMUNITIES AND TRANSIT AGENCY BUS STOP INVENTORIES

**Khoa Tran**
**Department of Computer Science and Engineering**
**University of South Florida**
**Tampa, FL 33620 USA**
**ktran9@mail.usf.edu**

**Edward Hillsman**
**Center for Urban Transportation Research**
**University of South Florida**
**Tampa, FL 33620 USA**
**hillsman@cutr.usf.edu**

**Sean Barbeau**
**Center for Urban Transportation Research**
**University of South Florida**
**Tampa, FL 33620 USA**
**barbeau@cutr.usf.edu**

**Miguel A. Labrador**
**Department of Computer Science and Engineering**
**University of South Florida**
**Tampa, FL 33620 USA**
**mlabrador@usf.edu**

## ABSTRACT

Proprietary formats and licensing restrictions limit individuals from sharing, viewing, or updating geographic (e.g., map) datasets and also restrict the use of these data to create innovative location-based services. Open geographic data repositories are now emerging that do not have these restrictions. The General Transit Feed Specification (GTFS) is a common open format for public transportation agencies' schedules, routes, and geographic bus stop information. However, transit agencies struggle to internally maintain and update these very large official datasets. Meanwhile, OpenStreetMap.org (OSM), an online free-content "Wikipedia-like" repository of geographic data, currently has little information about U.S. public transportation, but does have a large number of users willing to freely contribute their efforts to improve geographic data for their communities. This paper presents GO_Sync, a framework and open-source software tool for synchronizing transit data between the transit agency's official GTFS dataset and OSM. GO_Sync connects the wealth of data from GTFS datasets with the ability of the OSM community to contribute crowd-sourced improvements to large datasets. The GO_Sync tool therefore enables public transportation agencies to upload GTFS data to OSM and retrieve crowd-sourced data back, while the online community can edit and correct the bus stop locations and amenities based on existing GTFS data. Successfully translating GTFS data into the OSM

format will enable over 125 transit agencies across the U.S. to share their public-domain data with the OSM community. Improved bus stop inventories will result in more accurate and complete multi-modal trip planning and navigation information across car, bus, biking, and walking.

Keywords- Transit, Public Transportation, Bus stop, General Transit Feed Specification, GTFS, location-based services, OpenStreetMap, OSM.

## INTRODUCTION

Location-based services answer three questions: Where am I? What is around me? How do I get there [1]? As more individuals carry devices that are location-aware (via Global Positioning Systems (GPS), or cell network location), location-based services such as real-time navigation and location-based social networking have become increasingly popular. However, there are two primary obstacles for developing quality location-based services: inaccuracies in geographic datasets, and the availability of large open geographic datasets.

In order to provide quality location-based services, accurate and up-to-date geographic data is required. However, many errors can exist in datasets maintained by a single source, whether a company or public agency, as keeping large official datasets current requires sustained efforts to collect and update data as it changes over time. Bus stop inventories of transit agencies are not an exception. For example, the Travel Assistant Device (TAD), a real-time mobile phone application to aid transit riders with special needs in navigating the transit system, has experienced data quality issues when using transit data formatted in the General Transit Feed Specification (GTFS) [2]. GTFS is a common format for public transportation agencies' schedules, routes, and bus stops [3]. GTFS data are posted to transit agencies' websites in a zip file format and retrieved by many web and mobile applications, including Google Transit and Microsoft Bing. The TAD research team cites incorrect geo-coded bus stops, provided in a GTFS dataset from transit agencies, as one of the primary challenges in the operation of TAD [2]. Some of the stops were in the wrong location and, therefore, for these stops, TAD was not able to deliver its desired transit notifications to exit the bus at the correct place and time.

Another important aspect to developing quality location-based applications is data availability. Most geographic data (e.g., map data) are currently either locked into proprietary formats and systems or under licensing restrictions. In particular, Google, one of the most popular interactive online maps in the U.S., belongs to the commercial corporation Google Inc. Without permission from the owner, people cannot share, update, and add geographic data. In fact, Google does not share the map data underlying the Google Maps service. Consequently, innovations in areas such as location-based services are limited since developers cannot create new applications due to restriction of dataset availability and licensing agreements. To resolve both of the above problems, an open-source and open-data solution is needed; a solution that transit agencies, software developers, and online communities can use without limitations.

This paper presents GTFS-OSM Synchronization (GO_Sync), a framework to synchronize mapping contributions from online communities and transit agency bus stop inventories. GO_Sync enables a transit agency to easily share their GTFS datasets with online open-data

communities while tracking crowd-sourced improvements to the bus stop inventory so these contributions can be integrated with the agency's official GTFS dataset. OpenStreetMap, a "Wikipedia" for spatial information, is chosen as the online free-content repository of geographic data. GO_Sync is used to compare agency GTFS data and existing transit data in the online OSM database, and to detect and present differences between the two sources. Using the application programming interface (API) provided by OSM [4], GO_Sync then uploads bus stops into the OSM server, while crowd-sourced improvements to the inventory are presented to the transit agency for processing and integration into the official bus stop inventory. Successfully translating GTFS data into the OSM format will enable over 125 transit agencies across the U.S. to share their public-domain data with the OSM community. The GO_Sync framework is made available to the transit developer community as an open-source software application to assist in the open sharing and improvement of public transportation information.

## RELATED WORK

The OSM community provides several editing tools available to mapping contributors. Among those tools are Potlatch, JOSM, Vespucci, Mapzen, and ArcGIS editor for OpenStreetMap. Potlatch is a light weight online Flash-based editor which allows users with an OSM account to add, update, or delete geographical data through an easy-to-use interface [20]. Although Potlatch is currently the main editor for OSM site, it is only ideal for quick manual editing contributions. In the same manner, Vespucci and Mapzen are also the two OSM editors for Android and iPhone. They do not support a large import of geographic data into OSM. The two most popular editors that support a bulk upload are JOSM and ArcGIS editor for OpenStreetMap. These two editors are great tools for geographic analysts and advanced OSM users since they support editing of a locally stored dataset before uploading changes as a whole. However, neither JOSM nor ArcGIS editor for OpenStreetMap allows users to automatically analyze and display conflicting information between a large independent dataset and the OSM database. Additionally, none of the existing OSM editors focus on bus stop inventories in GTFS format. Therefore, it is difficult for transit agencies to either contribute their GTFS dataset to OSM or retrieve crowd-sourced improvements specific to their bus stop inventory. GO_Sync differs from the above existing OSM tools in that GO_Sync focuses on automated information analysis and synchronization between a transit agency and OSM based on the agency bus stop inventories. The targeted end-users of this framework are therefore public transit agencies, or transit information software developers working on behalf of public transit agencies. Using GO_Sync, they can visualize crowd-sourced improvements to bus stops, as well as which contributors are active and trusted. GO_Sync can also export OSM bus stop data in GTFS format (i.e., comma-separated values (CSV) files) that could be re-used for other routing software or integration into GTFS datasets.

## PROBLEM DEFINITION

The concern of this paper is primarily with the synchronization of data from GTFS format and the OSM repository. The primary spatial component of the GTFS dataset is the location and attributes of bus stops and bus stations, which are the data that GO_Sync processes. At first appearance, it seems that GO_Sync could simply do a bulk import of new GTFS data into OSM every time the transit agency generates a new GTFS dataset. However, GO_Sync cannot blindly

insert the entire GTFS dataset into OSM, as it would destroy the existing crowd-sourced efforts by other OSM users. Although there is a suggested coding system of tagging bus stop data in OSM, many OSM contributors do not follow these guidelines as OSM is an open-tagging collaborative system with users with various levels of OSM experience. Therefore, matching existing crowd-sourced work with new GTFS data programmatically is the most challenging issue in synchronizing GTFS data and OSM data. In addition, for all executions after the very first bulk upload of GTFS data from the transit agency, GO_Sync needs to compare the current contents of OSM against the GTFS data so that GO_Sync does not duplicate or overwrite the existing OSM data.

Solving the above problems requires several steps:

1) GO_Sync must determine what has already been mapped in the OSM in the area covered by the GTFS data.
2) GO_Sync must identify bus stops that may be in conflict between OSM and an agency's GTFS data.
3) GO_Sync must make decisions on how to manage the conflict, with the aid of user input.
4) GO_Sync uploads GTFS data into OSM, managing any conflict that the user cannot reconcile.

This process must be repeatable so that when the transit agency changes its bus service and needs to reflect this in its GTFS data, the agency can repeat the above four steps and benefit from any corrections made by individuals mapping in OSM.

## OPENSTREETMAP DATA

OSM exposes an application programming interface (API) for fetching and saving raw data to and from the OSM PostgreSQL database. The current version in use is API v0.6 [5]. This API is primarily based on RESTful API under the form of HTTP GET, PUT, POST, and DELETE requests, using HTTP basic authentication with a username and password [4]. Furthermore, the payload is a XML document using MIME type "text/xml" and UTF-8 character encoding.

Two OSM data primitives widely used in this application are nodes for representing bus stops and relations for representing bus routes. A node also consists of a geospatial point defined by latitude and longitude, along with other attributes. A relation has attributes to specify affiliation among OSM objects. A relation can have zero, one, or many nodes as its members. Specifically, GO_Sync determines which bus stops relate to which routes; then, it links all bus stops to corresponding OSM relations. Bus routes, therefore, are defined as properties of bus stops.

Since OSM is a collaborative mapping project, the data are created by different users from various backgrounds such as professional mappers, developers, and students. Therefore, OSM data employ an "open" tagging system, which does not enforce a standard set of tags. However, OSM does provide guidance for best-practices on tagging and recommended tags. Applications must take this open tagging system into account and recognize that tags with different names can have the same meaning. For instance, when mapping a bus stop, the bus route could be defined as

"route," "routes," "route_id," while the recommended tag is in fact, "route_ref." [6]. This problem becomes more challenging as we consider coding bus stops with multiple operators (i.e., transit agencies) at a single stop. The recommended tagging practice for OSM users and multiple operators is to input the key name as "operator" and use a semicolon as the delimiter to separate different operator names in the corresponding value field, but this practice is not always followed.

## Potential Bus Stop Conflicts

Managing bus stop inventories is a tremendous effort for transit agencies. This problem is made worse when public agencies with other jurisdictions (e.g., departments of transportation, universities, etc.) may move bus stops without the agency's knowledge during construction. Hence, agencies are not always able to keep a centralized database with current positions of bus stops.

The authors of this paper found several conflicts between the existing OSM stops and the GTFS file of Hillsborough Area Regional Transit (HART) in the Tampa, Florida area. In the context of GO_Sync, the term "conflict" refers to a stop that exists in the OSM database but does not have an associated GTFS ID, or an OSM stop with a geographic location that does not exactly match the location of a GTFS bus stop. In other words, a bus stop conflict involves at least two distinct stops nodes that may represent the same logical bus stop. Figure 1 shows a set of conflicting bus stops along 50th Street in Tampa, Florida, USA.

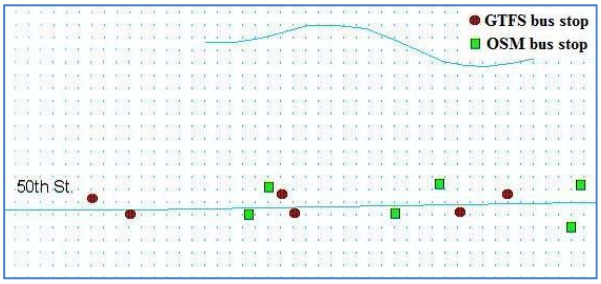

Figure 1. 50th Street conflicts

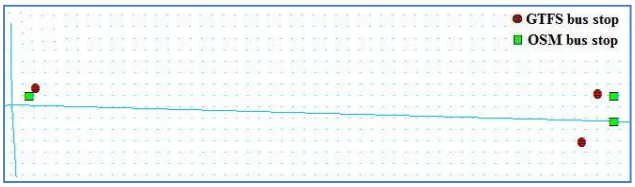

Figure 2. Holy Drive conflicts

The red dots represent GTFS stops while the green squares symbolize existing stops in OSM. Another conflict is captured on Holy Drive in Tampa, Florida, USA where GTFS data shows an unpredicted shift compared to OSM stops (Figure 2). Sometimes, stops that have been removed by the transit agency are still included in GTFS datasets (Figure 3).

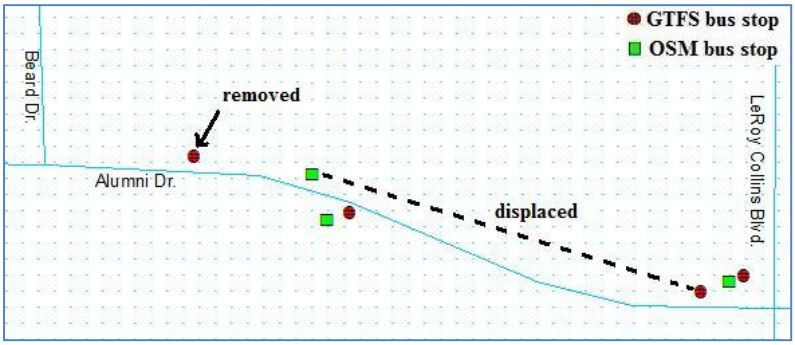

Figure 3. Alumni Drive conflicts

The stop with the "removed" label in Figure 3 still exists in the GTFS dataset but has been discontinued by the agency. Figure 3 also shows a stop that was displaced due to a nearby construction.

After the analysis of various conflicts with stops data in the University of South Florida area, a 400-meter threshold was chosen to associate potentially conflicting GTFS stops and OSM data. In other words, if there is an OSM stop within 400 meters of a GTFS stop, these stops need to be compared since they may represent the same logical stop.

As GO_Sync is a Java desktop application, a geospatial database is not involved. Therefore, the geodesic distances between nodes, defined as the shortest path along the ellipsoid of the earth at sea level, must be calculated within the application. By definition, the geodesic distance is the shortest path along the ellipsoid of the earth at sea level between two geographic coordinates. Over the years, a significant amount of work has been done to derive formulas that compute geodesic distance of two points [7][8][9][10]. The Vincenty inverse solution produces distance measurements that are accurate within 0.5mm [11].

Vincenty's formula follows:

a, b = major and minor semi axes of the ellipsoid
f, flattening = (a-b)/a
$\varphi_1, \varphi_2$ = geodesic latitude
L = difference in longitude
U1, reduced latitude, defined by tanU1 = (1–f)tan$\varphi_1$
U2, reduced latitude, defined by tanU2 = (1–f)tan$\varphi_2$

$$\lambda = L \quad (1)$$

$$\sin^2\sigma = (cosU_2 sin\lambda)^2 + (cosU_1 sinU_2 - sinU_1 cosU_2 cos\lambda)^2 \quad (2)$$

$$cos\sigma = sinU_1 sinU_2 + cosU_1 cosU_2 cos\lambda \quad (3)$$

$$tan\sigma = \frac{sin\sigma}{cos\sigma} \quad (4)$$

$$sin\alpha = cosU_1 cosU_2 \frac{sin\lambda}{sin\sigma} \quad (5)$$

$$cos2\sigma_m = cos\sigma - \frac{2sinU_1 sinU_2}{\cos^2\alpha} \quad (6)$$

$$C = \frac{f}{16}\cos^2\alpha[4 + f(4 - 3\cos^2\alpha)] \quad (7)$$

$$L = \lambda - (1 - C)fsin\alpha\{\sigma + Csin\sigma[cos2\sigma_m + Ccos\sigma(-1 + 2\cos^2 2\sigma_m)]\} \quad (8)$$

The procedure starts from Equation (2) through (6), and then a new $\lambda$ is obtained from Equations (7) and (8). The iteration repeats until the change in $\lambda$ is negligible. Afterward, the distance s is obtained. Distance is then computed between two coordinates on the Earth surface:

$$u^2 = \cos^2\alpha \frac{(a^2 - b^2)}{b^2} \quad (9)$$

$$A = 1 + \frac{u^2}{16384}\{4096 + u^2[-768 + u^2(320 - 175u^2)]\} \quad (10)$$

$$B = \frac{u^2}{1024}\{256 + u^2[-128 + u^2(74 - 47u^2)]\} \quad (11)$$

$$\Delta\sigma = Bsin\sigma\{cos2\sigma_m + \frac{B}{4}[cos\alpha(-1 + 2\cos^2 2\sigma_m) - \frac{B}{6}cos2\sigma_m(-3 + 4\sin^2\sigma)(-3 + 4\cos^2 2\sigma_m)]\} \quad (12)$$

$$s = bA(\sigma - \Delta\sigma) \quad (13)$$

The WGS 84 reference system is used to represent the earth in Global Positioning Systems [12]. Vincenty's formula parameters are therefore configured for WGS84 accordingly with *a=6378137; b=6356752.3142; f=1/298.257223563*.

## FRAMEWORK DESIGN

Two difficulties in constructing a communication tool between a transit agency and an online community are the unpredictable nature of tagging in OSM and the high potential for conflicts between GTFS dataset bus stops and existing bus stop information within OSM. Since GTFS is a well-designed data model that identifies bus stops with unique ids for each agency, the difficulties are thought to be easily overcome by removing all the conflicting OSM nodes and inserting new stops from the GTFS dataset. However, that method would violate the basic principle of open-source software and the social mapping model: respect others' works. Therefore, the framework must not damage crowd-sourced contributions to public transportation data in OSM. Instead, the framework should supplement the existing OSM data with the transit data from the GTFS dataset, while properly resolving potential conflicts between the two groups of data. Additionally, the framework should present crowd-sourced improvements of transit data to the transit agency so the agency can improve their official bus stop inventory.

The GO_Sync framework starts by processing a new GTFS dataset. Using the 'stops.txt,' 'trips.txt,' 'routes.txt,' and 'stops_times.txt,' all agency's stops and associated routes are read into the software tool. To retrieve bus stops and routes that have been previously coded in OSM, GO_Sync searches the input GTFS dataset to find the maximum and minimum of both latitude and longitude among all the bus stop coordinates. A bounding box is created using those values. To reduce the payload size downloaded from the server, GO_Sync only retrieves OSM primitive nodes coded with tag *[highway=bus_stop]* and relations coded with tag *[route=bus]*. Since the main API does not support a straightforward method to query data from OSM server, GO_Sync uses the read-only extended API (XAPI). The response from XAPI server consists of all bus stops and routes that satisfied our criteria as a XML document. After processing this response stream, all OSM bus stops and routes in the described bounding box, along with their associated tags, are collected into corresponding lists for further calculations.

### Bus Stop Comparisons

Bus stops in OSM that are in close proximity to stops in GTFS datasets are potentially duplicates of the GTFS stops; conversely, GTFS that are in close proximity to stops in OSM are potentially duplicates of the OSM stops. The framework presented in this paper attempts to resolve these "conflicts."

The GTFS OSM Synchronization (GO_Sync) framework first retrieves OSM bus stops which have the same operator name of the transit agency (e.g., "Hillsborough Area Regional Transit") selected within the tool or where the operator field is null, or missing are retrieved from OSM. Bus stops are identified by their operator names and the gtfs_id fields. In some cases where the operator names and gtfs_id fields are not present in OSM, GO_Sync uses the geographic latitude and longitude to make distinctions among stops. A bus stop object in GO_Sync is instantiated with an operator name, bus stop ID from GTFS (i.e. gtfs_id), latitude, longitude, and additional data. In the case of a stop from OSM that does not have a gtfs_id, the field is recorded as 'none.' When inserting into OSM, bus stops may also include other useful information from GTFS data such as stop code, description of a stop, fare zone for a stop, etc. The transit agencies can also receive extra information from OSM users such as the existence of bus shelters, benches, lighting facilities, as well as if maintenance is needed at specific bus stops. In general, the goal of the GO_Sync framework is to build a communication bridge between formatted GTFS data and OSM online community by categorizing bus stops into four different groups:

1) New GTFS stops having no conflict with stops in OSM
2) New GTFS stops having possible conflicts with stops in OSM
3) GTFS stops which already exist in OSM but have updated information from the agency, and
4) GTFS stops which are identical to OSM stops.

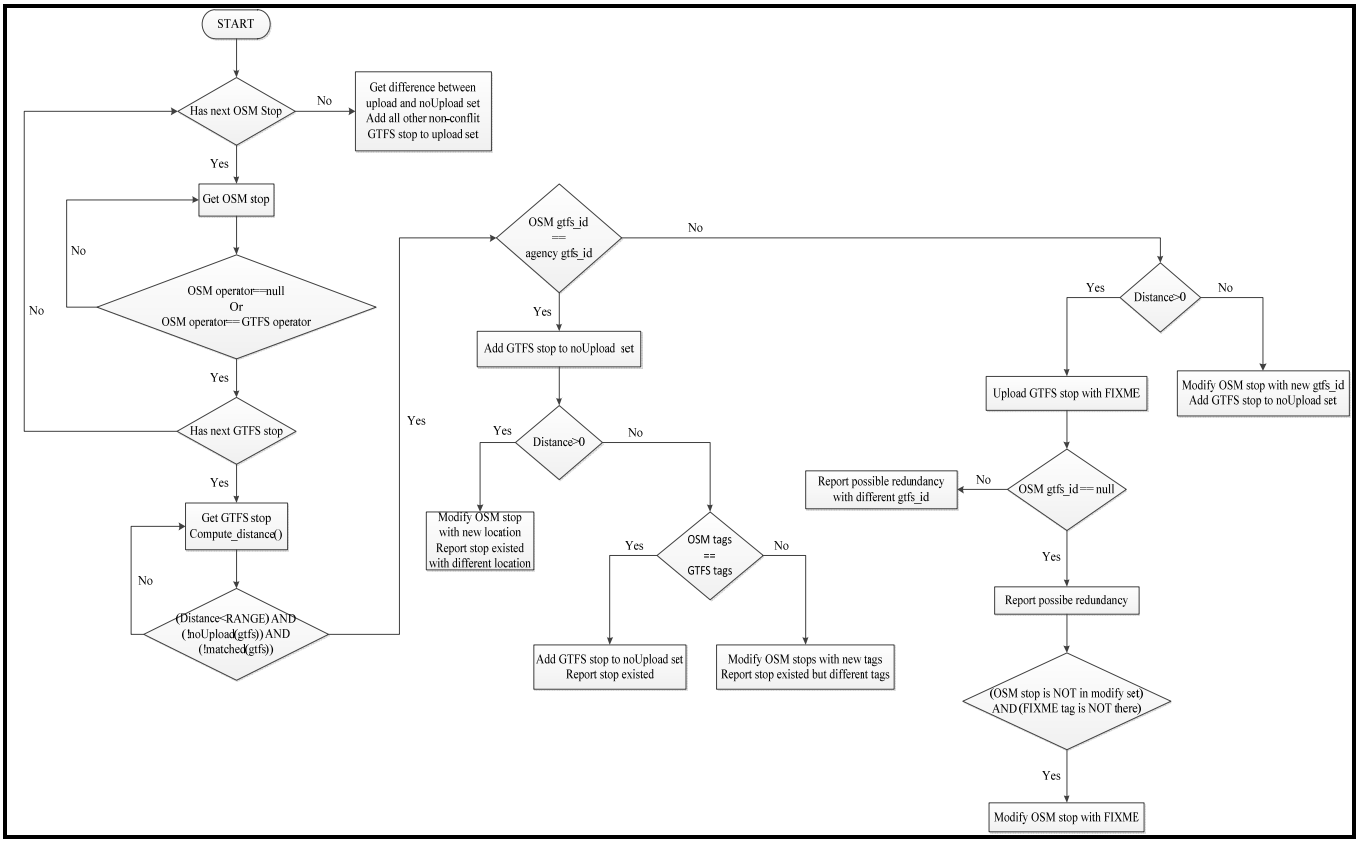


Figure 4. The GO_Sync Framework Algorithm

The algorithm starts after reading data from an input GTFS dataset and downloading copies of the bus stop information stored in OSM. Items in the GTFS data and OSM collection are then compared in pairs to determine which of the four above groups they belong to (Figure 4). A "FIXME" tag is also added to the stop to notify other OSM users about the update to the stop. One of the main parameters in this algorithm is the distance used to verify redundancy between the new stops and existing ones in the OSM server. Currently, that factor is set to 400 meters, because errors of this size have been observed in existing GTFS data. Once the GO_Sync algorithm finishes the classification stage, a graphical user interface report is displayed to allow the end-user to review what data are going to be uploaded, as well as the inconsistencies between their provided GTFS input and existing OSM data (Figure 5).

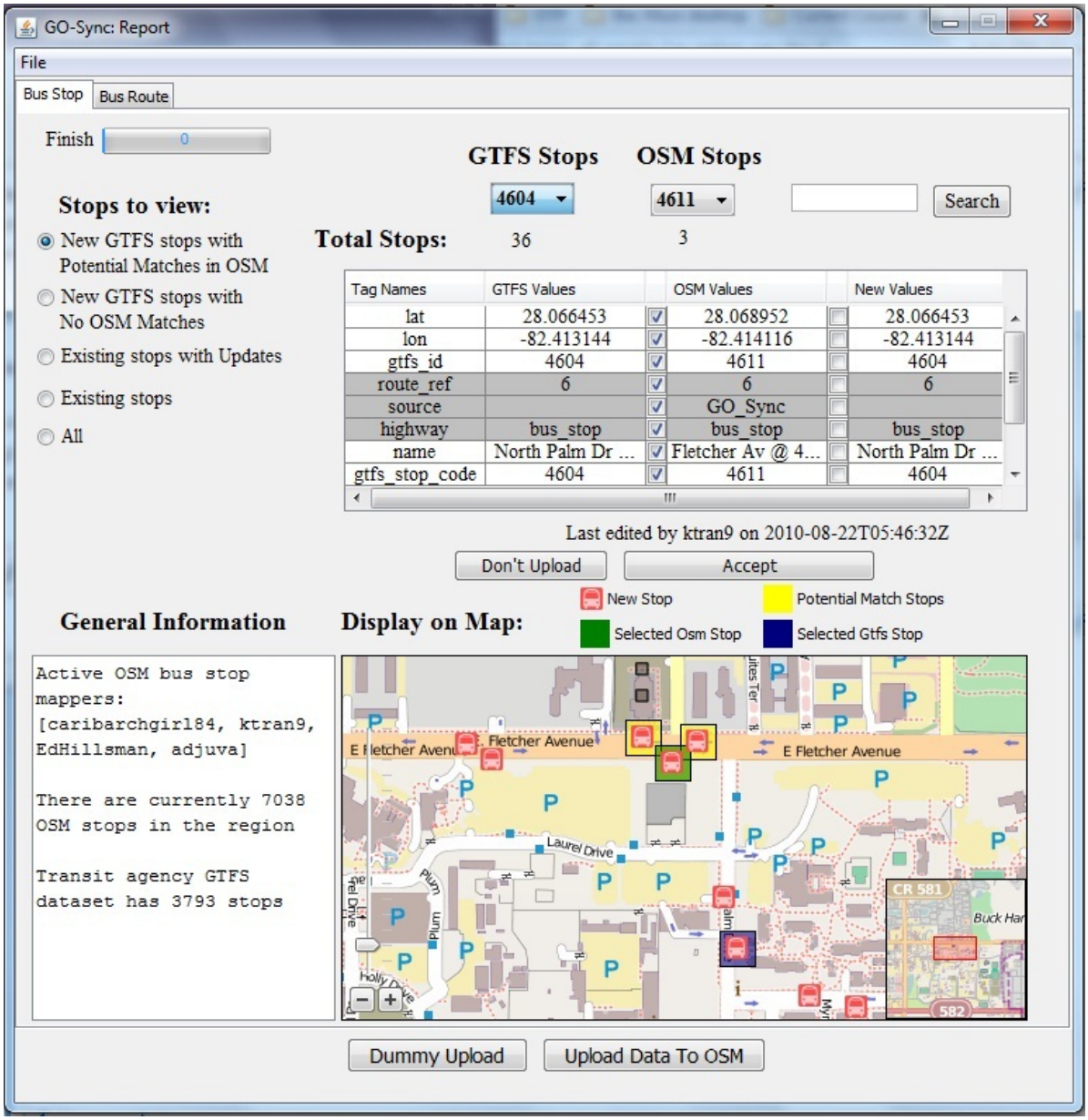


Figure 5. The user interface of the GO_Sync Framework which shows the differences between General Transit Feed Specification datasets and bus stops in OpenStreetMap

## Bus Stop Tag Comparisons

In OSM, a tag is a 2-tuple (i.e. ordered pair) in which the first element is often called "key" and the second element is regarded as "value." OSM does not place any restrictions on tags but does not allow duplicate keys in the same node, way, or relation. Tags are also widely used by OSM users to label nodes and provide supplementary data for corresponding objects. GO_Sync takes this social mapping model into account and determines which tags are new, which tags already exist in the server, and also detects any changes in tag values. GO_Sync performs these comparisons with existing OSM tags and adds the information included in GTFS input.

## Bus Route Comparisons

Transit agency bus route data are generated through the use of OSM relations. In practice, an OSM relation can only be linked with existing objects. Therefore, of the four total bus stop groups, GO_Sync takes into account bus routes from only two bus stop groups: stops which exist in OSM but have updated information from the agency, and stops which are identical between the two datasets. Those bus stops have been previously coded in the server and, therefore, have their own OSM ID.

Bus routes are identified by their operator names and route_ref (i.e., route number). Unlike bus stops, when the route_ref field is empty, GO_Sync simply creates a new relation without any further considerations.

The process begins with constructing new route objects for all bus stops that already have an OSM ID. All stops are linked as relation's members accordingly. After getting OSM relation data from the specified bounding box, the algorithm compares all routes to detect if any relations need to be modified. This method is very similar to the tag comparison routine but with one extra evaluation for bus stop members. After this procedure is complete, all the routes are categorized into three different groups: new, modified, and routes that are already in OSM and do not have any new updates from the agency.

## Modifications to OSM Data

### Bulk upload

When the GO_Sync framework finishes classifying the four sets and the application receives the upload command from the end-user, it is ready to apply modifications to the OSM server. While GO_Sync gets OSM bus stops data from the XAPI server, the application uploads with the main API database. Appropriate data are written into an output stream to the OSM API database as follows:

1) ***Create:*** new bus stops for all elements in the two sets *'new GTFS stops having no conflicts'*, and *'new GTFS stops with possible conflicts'* are created. In addition, new bus routes for all objects in the new route set are created. Appropriate tags and corresponding values are also added to the OSM server.
2) ***Modify:*** Stops for all elements in the set *'GTFS stops which already exist in OSM but have updated information from the agency'* are modified. Analogously, all routes in the modified set are modified. Proper tags are also inserted to each Stop and Route object.
3) ***No action:*** No action is taken for the stops and routes that are identical between the two datasets.

The data flow diagram of this application is presented in Figure 6.

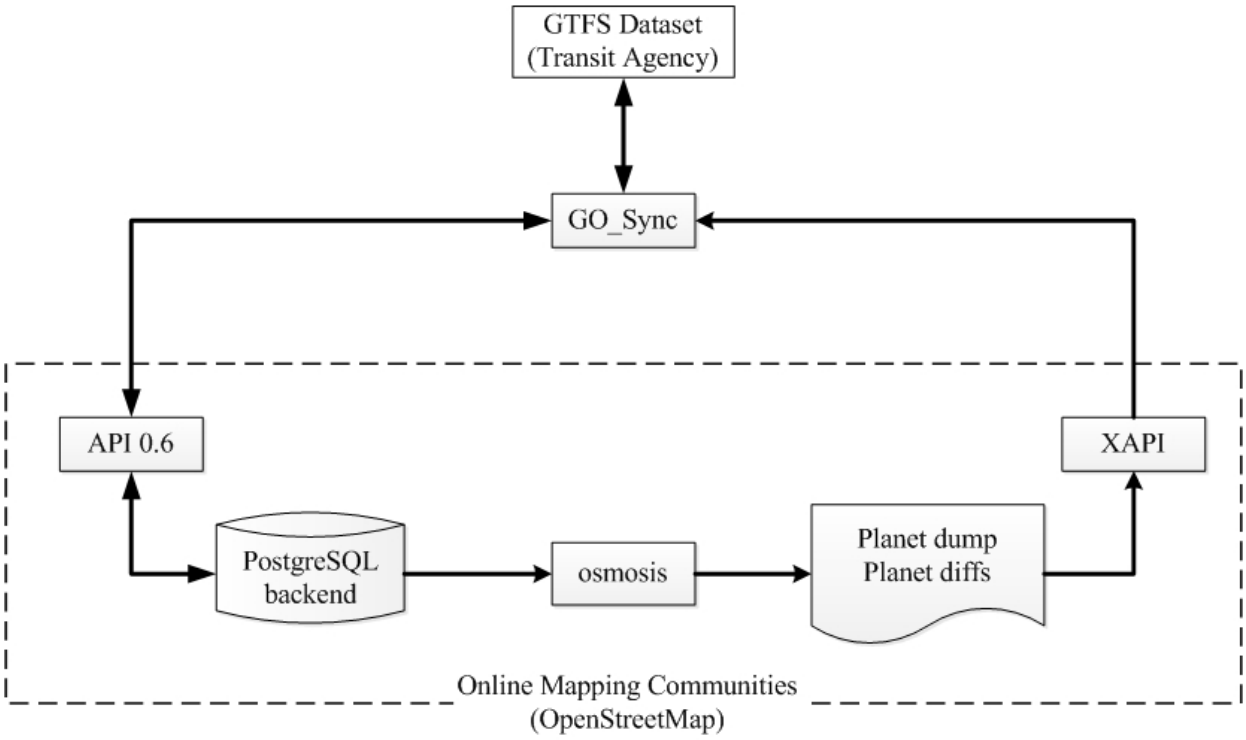


Figure 6. The GO_Sync's Data Flow Diagram.

Any edits in OSM are done within a changeset. Therefore, a new changeset must be created before importing any data into the OSM server. Additionally, uploading large datasets into the OSM server requires that the data be in OsmChange file format, which is used by Osmosis, a command line Java application for processing OSM data, to describe differences between two dumps of OSM data [13][14]. In the context of GO_Sync, once the bus stop record is successfully uploaded to the OSM PostgreSQL database, Osmosis would apply the changes into the "planet dump," which performs an update every minute from the database. In fact, planet dump is the raw data for many OSM services, including the XAPI. Consequently, there is a time-lag resulting in differences between PostgreSQL database and XAPI database. Thus, after uploading new data to OSM, a GO_Sync user should wait a short period of time before running GO_Sync again to ensure that enough time is given for the updates to appear in OSM.

Revert changeset

As mentioned earlier, a changeset is an OSM object which contains all user activities when they interact with the server for uploading new dataset. Occasionally, a GO_Sync user may make mistakes in importing stops into OSM. As a result, the user may want to reset all changes that they have made into earlier states. If no changes have been made by other users to the same data, the changeset is "fresh." However, if other users have edited the data, the changeset is "soiled."

GO_Sync provides users a convenient way for correcting their mistakes. Two limitations of this implementation are that the revert only takes place on bus stops (e.g., changeset created by GO_Sync) and the changeset must be 'fresh.'

Initially, users are asked to input a changeset id that they want to revert. Based on the changeset ID, all changes that have been made in that specific changeset are obtained. There are three possible actions that could have occurred within a changeset, which GO_Sync can reverse. These three actions, and the corresponding reversal, are:

1) ***Create:*** GO_Sync simply deletes that bus stop.
2) ***Modify:*** GO_Sync obtains the OSM ID and the current version of that specific node. Then, the application retrieves the previous state of the node. Next, GO_Sync modifies the node with corresponding information. The three different attributes from its previous state after the reversion are the creator name, version number, and timestamp value.
3) ***Delete:*** Similar to the modify process. However, instead of modifying nodes, GO_Sync creates a new node on the server with the retrieved data. Again, the creator name, version number, and timestamp value cannot be restored.

## Sample Execution

The main end-user of this application is the transit agency, or transit software developers working on behalf of the transit agency. Public transportation schedules and associated geographic information are provided with the GTFS format. The user first needs to sign up for a free OSM account. When GO_Sync is started, it asks the end-user for their OSM username and password, as well as a description to include with the changeset when it is uploaded to OSM. The user then chooses whether he or she wants to analyze bus stop data or revert a changeset.

### Scenario 1 – Analyze Bus Stop Data

Comparing bus data requires the user to enter the agency full name, possible alias, GTFS source file, and the operator ID from the National Transit Database. When the agency finishes inputting information, the application defines a bounding box which includes all new GTFS bus stop locations. GO_Sync then downloads OSM bus stops and routes in the described bounding box and stores them into a set of stop objects and a set of route objects accordingly. Next, the stop comparison algorithm executes to provide information about which bus stops and associated properties should be uploaded. During this process, the tag comparison algorithm classifies the nodes. When the stop comparison algorithm finishes, the route comparison is then invoked. Then, all stops and routes are categorized in their corresponding groups. A report is generated for the end-user to review before new or modified data are to be inserted into the OSM database. The report is an interactive graphical user interface (GUI). The GUI helps users to manage bus stop inventories using an online map. Users can remove bus stops, edit and add tags, as well as re-classify stops and routes into different groups. When a user finishes editing the dataset, they can continue to upload their data into OSM.

### Scenario 2 – Revert a Changeset

This task only requires one input field, which is the changeset ID. After the changeset ID is filled in, GO_Sync downloads the entire changeset. It then processes the XML data and categorizes the data into *create*, *modify*, and *delete* sets. For every created node, GO_Sync deletes the nodes

while nodes in other two sets need to be retrieved one by one. The time to process all the nodes depends primarily on the number of elements in the *modify* and *delete* sets.

## EVALUATION

The GO_Sync application was installed in a personal computer with an Intel Core 2 Duo CPU T5750, 4GB RAM, and a 64-bit operating system. Since GO_Sync requires a Java Virtual Machine, the latest Java Runtime Environment (JRE) 6 was also installed in the same computer [15]. The authors obtained permission from Hillsborough Area Regional Transit Authority (HART), located in Tampa, FL, US, to upload their GTFS data into OSM.

Before the GO_Sync tool was executed using HART data, little public transportation information existed in the Tampa, FL, US area even though HART serves nearly 1.2 million people [18]. Only 133 HART bus stops were available in OSM prior to running GO_Sync (Figure 7). After the authors received HART's approval to use its GTFS data, in July 2010 GO_Sync uploaded 3812 new bus stops to OSM (Figure 8). The trend of a lack of large amounts of public transportation data in OSM is similar across the U.S., since OSM originally started in Europe in August 2004 and it wasn't until September 2007 that the Topologically Integrated Geographic Encoding and Referencing system (TIGER) data for US streets, produced by the US Census Bureau, was uploaded to OSM [17]. Therefore, OSM is still relatively young as a major geographic data repository in the U.S.

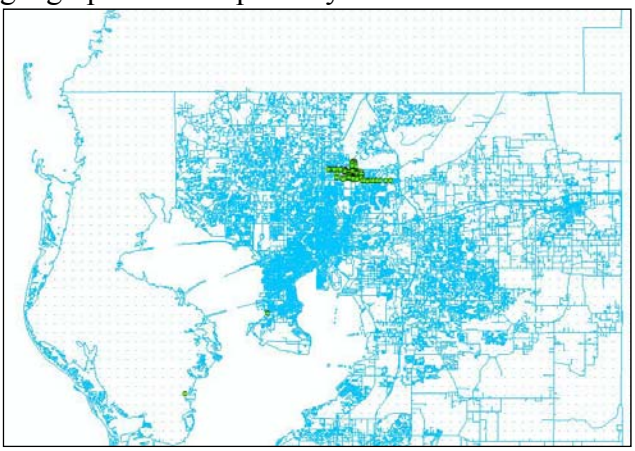

Figure 7. Bus stop data for Hillsborough County in OSM before using GO_Sync.

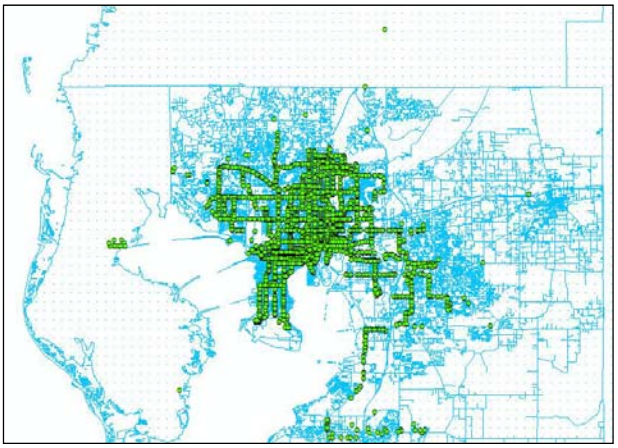

Figure 8. Hillsborough County bus stop data in OSM after using GO_Sync.

In order to determine the activity of OSM users as related to public transportation data, the authors analyzed the OSM transit data for the Tampa area again in January 2011. Between July 2010 and January 2011, there were modifications to the geographic location of 173 bus stops by OSM users, including 4 unique OSM users other than the authors. Figure 9 shows the distribution histogram of the distance that bus stops were moved by OSM users.

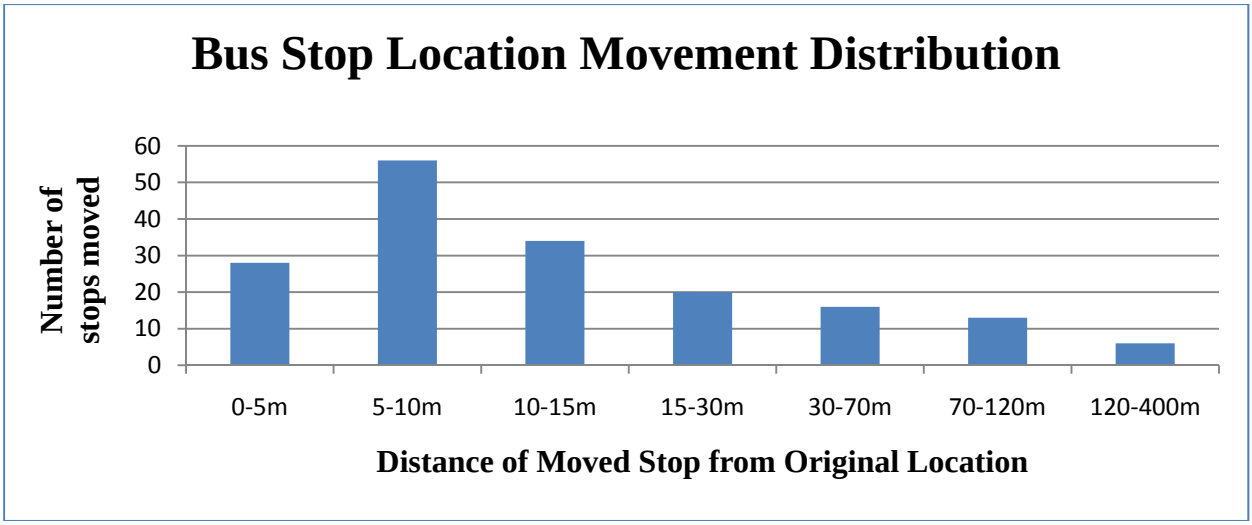


Figure 9. OpenStreetMap users have modified the geographic location of 173 bus stops in the Tampa, Florida area

The authors reviewed a subset of the modified bus stop locations and confirmed that many of the moved stops in OSM were indeed corrections to the initial inaccurate geographic location of the bus stop as it was represented in the GTFS dataset. The authors also confirmed with a representative of HART that the GTFS bus stop latitudes and longitudes are approximated and would not necessarily reflect the true geographic position of the stop.

Based on these findings, transit agencies should be able to benefit from these crowd-sourced modifications in order to improve a bus stop inventory without significant cost to the agency. GO_Sync assists transit agencies in this task by generating a report for the transit agencies which identifies the bus stops that have been modified by OSM users (Figure 5). Ideally, active OSM users will continue to update information about local stops so that the transit agency would be able to periodically run GO_Sync to upload new GTFS datasets, and in the process receive additional improvements from OSM users to bus stop locations and amenities. GO_Sync is designed according to a social mapping model to ease the data management and maintenance efforts of a transit agency for their transit data. GO_Sync can be downloaded by developers and users at http://code.google.com/p/gtfs-osm-sync/ [21].

## CONCLUSION

Open data sets such as the General Transit Feed Specification (GTFS) and OpenStreetMap (OSM) are transforming the landscape of information systems for public transportation and are leading to innovations in mapping and location-based services. While transit agencies struggle to keep large bus stop inventory datasets up-to-date, the OSM community has thousands of users that are willing to contribution improvements to geographic data. However, in the U.S. there is currently little public transportation information in OSM. The GO_Sync framework presented in this paper is able to synchronize large GTFS datasets from transit agencies with crowd-sourced OSM data. GO_Sync is therefore able to contribute large amounts of public transportation data to OSM while helping transit agencies maintain and update their inventories by leveraging crowd-sourced improvements to the data. A test deployment in Tampa, FL has yielded improvements to

173 bus stop locations by OSM users. The authors also make GO_Sync available to the open-source transit developer community as an open-source application, which is available at http://code.google.com/p/gtfs-osm-sync/. Future research will focus on further evaluation of this tool, additional improvements to better fit transit agency's needs, as well as determining the ideal distance used to identify conflicting GTFS and OSM bus stops.

## ACKNOWLEDGEMENT

This work was partially supported by the National Center for Transit Research and the Florida Department of Transportation under grant TWO 977-20 "Enabling Cost-Effective Multimodal Trip Planners through Open Source Transit Data" and the NSF Grant No. 0754537 "An REU Site in Computer Science and Engineering."

## REFERENCES

[1] Q. H. Mahmoud. (2004, March). J2ME and Location-based Services [Online]. Available: http://developers.sun.com/mobility/apis/articles/location

[2] S.J. Barbeau, M. A. Labrador, N. L. Georggi, P. L. Winters, R. A. Perez. "The Travel Assistance Device: Utilizing GPS-enabled Mobile Phones to Aid Transit Riders with Special Needs," Institution of Engineering and Technology (IET) Intelligent Transportation Systems, 2010, Vol. 4, Issue 1, pp. 12–23. doi: 10.1049/iet-its.2009.0028. © The Institution of Engineering and Technology 2010.

[3] "General Transit Feed Specification – Google Code." Internet: http://code.google.com/transit/spec/transit_feed_specification.html, January 11, 2010 [July 27, 2010]

[4] "API v0.6 – OpenStreetMap Wiki." Internet: http://wiki.openstreetmap.org/wiki/API_v0.6, June 1, 2010 [July 27, 2010]

[5] "API – OpenStreetMap Wiki." Internet: http://wiki.openstreetmap. org/ wiki/API, June 24, 2010 [July 27, 2010]

[6] "Tag: highway=bus_stop – OpenStreetMap Wiki." Internet: http://wiki.openstreetmap.org/wiki/Tag:highway%3Dbus_stop, July 23, 2010 [July 27, 2010]

[7] Rainsford, H. F. (1955). "Long geodesics on the ellipsoid." Bull. Geod., 37, 12–21.

[8] Kivioja, L. A. (1971). "Computation of geodetic direct and indirect problems by computers accumulating increments from geodetic line elements." Bull. Geod., 99, 55–63.

[9] Vincenty, T. (1975). "Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations." Surv. Rev., XXII (176), 88–93. [Online]. Available: http://www.ngs.noaa.gov/PUBS_ LIB/ inverse.pdf

[10] Bowring, B. R. (1981). "The direct and inverse problems for short geodesic lines on the ellipsoid." Surveying and Mapping, 41(2), 135–141.

[11] C. M. Thomas, and W. E. Featherstone, "Validation of Vincenty's Formulas for the Geodesic Using a New Fourth-Order Extension of Kivioja's Formula," Journal of Surveying Engineering ASCE, February 2005. [Online]. Available: http://www.cage.curtin.edu.au/ ~will/thomas-featherstone.pdf

[12] "World Geodetic System." Internet: https://www1.nga.mil/Products Services/Geodesy Geophysics/WorldGeodeticSystem/Pages/default.aspx, August 4, 2008 [July 27, 2010]

[13] “OsmChange – OpenStreetMap Wiki.” Internet: http://wiki.openstreetmap.org/wiki/OsmChange, May 5, 2010 [July 27, 2010]
[14] “Osmosis – OpenStreetMap Wiki.” Internet: http://wiki.openstreetmap.org/wiki/Osmosis, June 26, 2010 [July 27, 2010]
[15] “Java SE 6 Documentation.” Internet: http://download.oracle.com/docs/cd/E17409 01/javase/6/docs/ [July 27, 2010]
[16] “HART – Hillsborough Area Regional Transit.” Internet: http://www.gohart.org/ [July 27, 2010]
[17] “History – OpenStreetMap Wiki.” Internet: http://wiki.openstreetmap.org/wiki/History, July 5, 2010 [July 27, 2010]
[18] “Hillsborough County QuickFacts from the US Census Bureau.” Internet: http://quickfacts.census.gov/qfd/states/12/12057.html/, April 22, 2010 [July 27, 2010]
[19] D.J. Cowen, S. Ahearn, M. Byrne. “The Changing Geospatial Landscape,” National Geospatial Advisory Committee, January 2009 [Online]. Available: http://www.fgdc.gov/ngac/NGAC%20Report%20-%20The%20Changing%20Geospatial%20Landscape.pdf/
[20] “Potlatch – OpenStreetMap Wiki.” Internet: http://wiki.openstreetmap.org/wiki/Potlatch December 6, 2010 [ January 20, 2010]
[21] “gtfs-osm-sync – General Transit Feed Specification (GTFS) OpenStreetMap SYNChronization (GO_SYNC)”, Internet: http://code.google.com/p/gtfs-osm-sync/, January 28, 2011 [January 28, 2011].