"TEAMTRACKER" - An Innovative Team Collaboration System

Sasa Savic and Hao Shi

School of Engineering and Science, Victoria University, Melbourne, Australia
sasa.savic@live.vu.edu.au and hao.shi@vu.edu.au

ABSTRACT

GPS (Global Positioning System) has been around for many years and becomes currently the most widely used system in determining position and measuring distance of objects on earth . Integration of GPS receivers in mobile phones and other portable devices has provided a powerful tool for developers and consumers of GPS enabled product to be utilized in number of ways. Coupling GPS with Wi-Fi and 3G enabled devices yield an even greater potential for integration and innovation. In this paper, "TeamTracker" is a proposed team collaboration system that is created by a framework of components utilizing GPS, Internet connectivity and web services. It communicates in sync to provide users with a set of platform and device independent tracking and collaboration features. This paper also aims to further examine system architecture, components and Google API. This project exhibits an enormous potential for many individual applications range from search and rescue, archeological exploration, agricultural analysis, fleet tracking or general team collaboration.

KEYWORDS

.NET, GPS, TeamTracker, WiFi, 3G, WebServices, Tracking

1. INTRODUCTION

The last decade has seen an exponential growth in many technology categories, and one of these is the mobile industry which certainly stands out and gains an increasing momentum due to its small and compact nature as well as the ability to integrate many components in one and provide a platform for development of wide applications. From the early Ericsson T-series that a lot of people used to refer to as "the brick", which only supported GSM interface, to the current iPhone and HTC ranges that incorporate numerous connectivity interfaces such as GPRS, 3G, HSDPA, Wi-Fi, Bluetooth and GPS – one can certainly see the shift in paradigm.

The introduction of many components that are being integrated in mobile phone devices allows application developers to interface devices with one another, allowing rich featured applications that suddenly become very useful in many aspects. An example of a location-based approach could be whilst GPS might be great for locating object or person's position – using the very same GPS data and sending it out to the internet by utilizing the mobile device Wi-Fi or 3G interface a lot more useful in other applications. It is certain that location-based applications take advantage of many interface features provided by the mobile phone vendors, so the focus of this paper is first to briefly describe Microsoft .NET's Windows Mobile SDK Library **[1]** that provides a platform for the development of windows mobile based applications. Then the paper presents an

DOI : 10.5121/ijcnc.2010.2507

insight of GPS and Internet connectivity and web services that are consumed by a mobile device running Windows Mobile Operating System.

Although the proposed "TeamTracker" framework provides a platform and device independent interface for team collaboration and tracking, the paper presents a detailed description and implementation of individual components using .NET Framework, PHP and WSDL (Web Services Definition Language).

2. RELATED WORK

In past few years several research topics have been put forward which are related to the abstract topics of wireless tracking systems. In this section, we explore these systems and reveal their pseudo-nature.

GlobeTrotter is a mobile emulator based system which was developed to overcome the difficulties of changing maps and the charges that users needed to pay in order to access the latest information [5]. The developed GlobeTrotter and GlobeTracker [6] systems are based on J2ME architecture and its components. Both systems utilize GPS and Google Maps as means of showing current location of an object or user.

It utilizes a server library, J2MEMAP on the 8motions server [7]. The systems were build on J2ME platform [8], that is amongst the most ubiquitous application platforms currently out there. Although the J2ME is a strong platform, recent analysis of mobile and embedded device development [9, 10] state that J2ME lacks in performance, whereas the .NET Compact Framework, being the chosen environment for the development of TeamTracker, manages to outweigh J2ME. The .NET CE platform also has greater platform independence and remarkable library interoperability support. This allows developers to utilize 3rd party COM Interoperability to derive greater functionality .

Another disadvantage of using J2ME platform is that it has no XML parsing or Web Service consumption support. This does not allow mobile devices to be as robust and collaborative feature rich. Custom libraries would need to be developed which yields in development time to be exponentially increased.

The system is primitive in comparison to current applications that seamlessly incorporate location mapping such as HTC and iPhone mobiles. The GlobeTrotter system, as illustrated in Figure 3 reads GPS data delivered by the GPS receiver. The longitude and latitude is then converted to decimal format accepted by all map hosting servers and sent to the 8motion server as a HTTP Web Request. The server requests map data based on given parameters and sends it back to the user, rendering their screen with current position data.

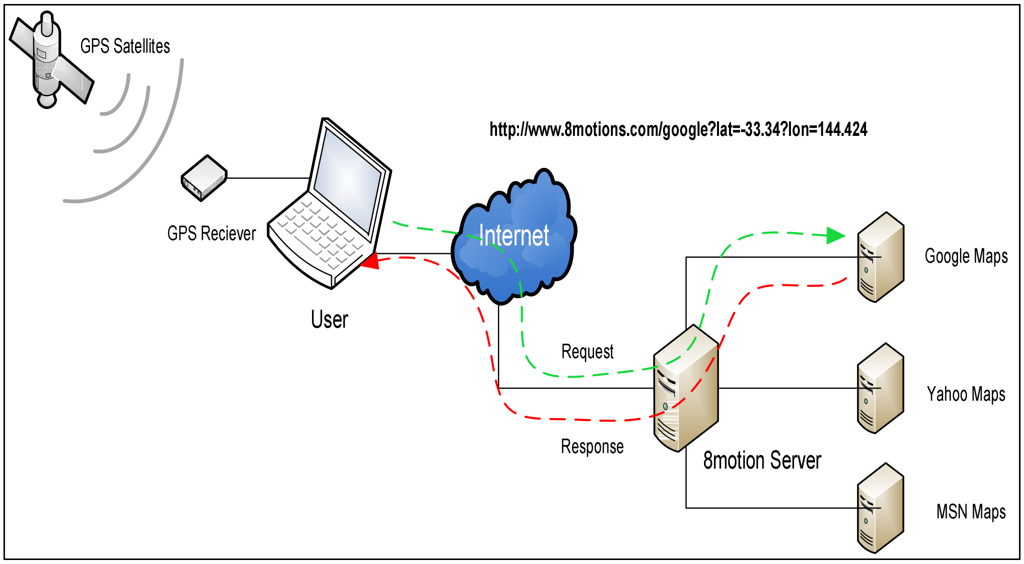


Figure 3. Functionality of GlobeTrotter [5]

This renders the system cliché, as the technologies have been implemented in ubiquitous mobile platforms. The systems underline the basic GPS protocols that encapsulate position data that, in turn, is provided to the mapping servers. The mapping is only achieved through static maps that take considerable amount of time to download, as each iteration of the position on the map needs a new request to the server to fetch map data.

TeamTracker ability to consume web services allows further integration and function sustainability over heterogeneous networks.

3. MICROSOFT .NET AND WINDOWS MOBILE SDK

Microsoft's .NET Framework and IDE tools such as the Microsoft Visual Studio allow developers to apply common skills across a variety of devices, application types and programming tasks **[2]**. It enables integration with other tools and technologies to build solutions. Microsoft .NET framework is an Object Oriented environment that consists of a Common Language Runtime (CLR) providing abstraction layer over the operating system. Developers can also consume a set of pre-built Base Class Libraries (BCL) for common low-level programming tasks. Being an Object Oriented type environment, it supports reusable and customizable solutions.

In order to support flexibility demanded by mobile market, Microsoft introduced Windows Mobile SDK that allows development on Windows Mobile and Windows CE type smart devices on its Windows Mobile compact mobile operating system. It provides developers with .NET Compact Framework Class Library. A high level architectural diagram is shown in Figure 1.

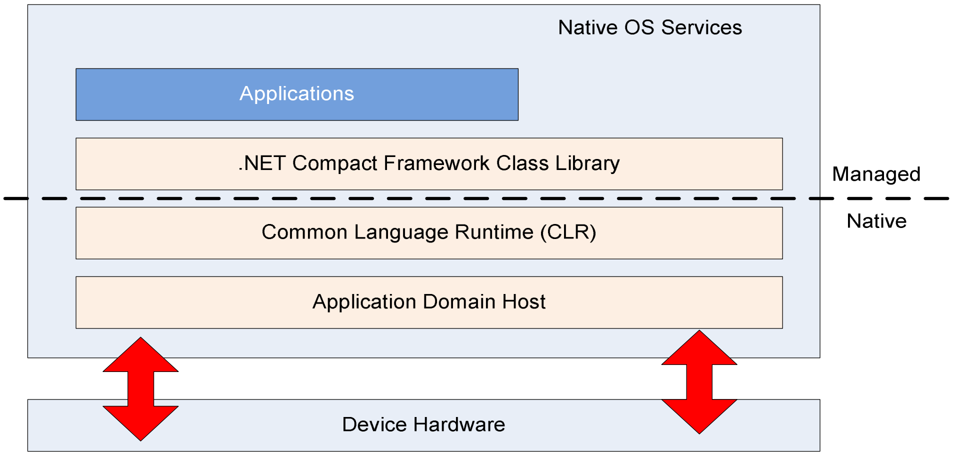


Figure 1. Architecture of Compact Framework [3]

The .NET Compact Framework Class Library is a set of API's **[3]** provided for building mobile applications. Some key class libraries utilized in "TeamTracker" are:

- Windows Form – for developing windows client application
- Base Classes – supporting advanced features such as threading, networking resources
- Data and XML – allowing easy handling of data and XML content
- Web Services – enabling developers of web service clients but not supporting web services as a service under the .NET Compact Framework.

4. Wi-Fi and 3G Internet Connectivity

Wi-Fi or 3G connectivity enables developers to create intuitive and function rich mobile applications capable of connecting to external services hosted around the world. One type of application that exhibits these features is iPhone's YouTube application. It is capable of consuming YouTube's web services that feed back relevant content based on user queries. It is also able to stream media content relevant to user search.

This feature is possible by exploiting the interfaces available to the device. If a particular user is connected to a Wi-Fi hotspot, the mobile device operating system is smart enough to determine the best way of connecting to the content requested by the user. If Wi-Fi connectivity is not available, the device attempts to fetch the same content by other means of communication, usually initiating a 3G data connection.

This is all seamlessly achieved by the underline operating system in the mobile device and developers who are capable of utilizing these functions by using the tools provided by the development platforms such as Windows Mobile SDK or Apple's Xcode iPhone Development to achieve this, can create very powerful applications. The proposed "TeamTracker" client application takes advantage of the interfaces and allows users to connect to external web services hosted in the internet cloud, from anywhere in the world.

5. Web Services and Web Services Definition Language (WSDL)

Web services have become increasingly popular as application scalability and reusability becomes an important factor for many application designs. Web services may reside in an intranet or internet environment, providing an abstracted mechanism of data exchange for thin or thick client applications. They are defined as web application programming interfaces that can be accessed over a network. A basic model of web service architecture is illustrated in Figure 2.

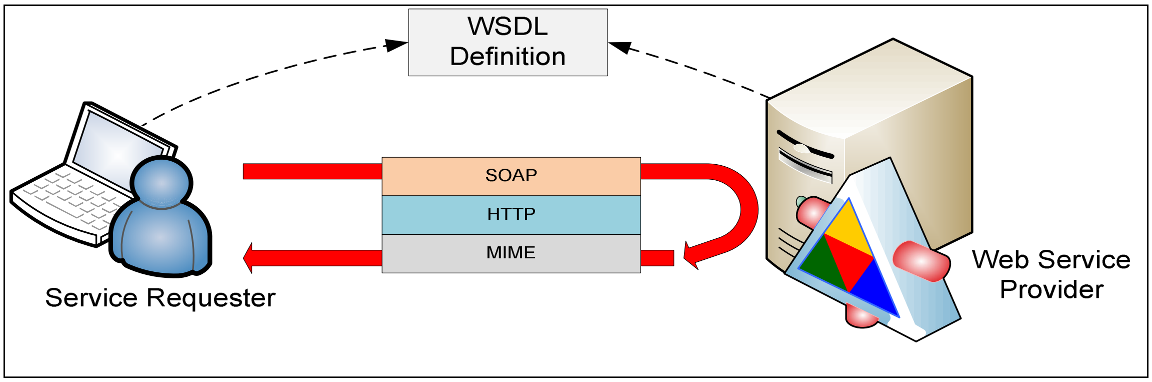


Figure 2. Web services architecture

Web services are defined in an XML format that describes the network services as set of endpoints or ports. In WSDL, the abstract definition of endpoints and messages (functions and functions parameters) are separated from their concrete network deployment or data format bindings **[4].** This model allows reusability of abstract definitions. The definitions can therefore bind to any protocol definition, although the three main communication protocols used in industry are SOAP 1.1 (Simple Object Access Protocol), HTTP's GET/POST methods and MIME (Multipurpose Internet Mail Extensions).

6. System Configuration

The implementation of "TeamTracker" system required following components:

- Microsoft Visual Studio 2008, C# Development Environment
- Microsoft Windows Mobile 6 SDK Library
- Web server running PHP 5.2 or above and MySQL database backend

7. Design and Development of TeamTracker

The "TeamTracker" system as illustrated in ***Figure 4*** further extends on the ideas of GlobeTrotter and GlobeTracker by implementing a framework of components that consist of client application, exposed web services and a logic layer that processes web service requests.

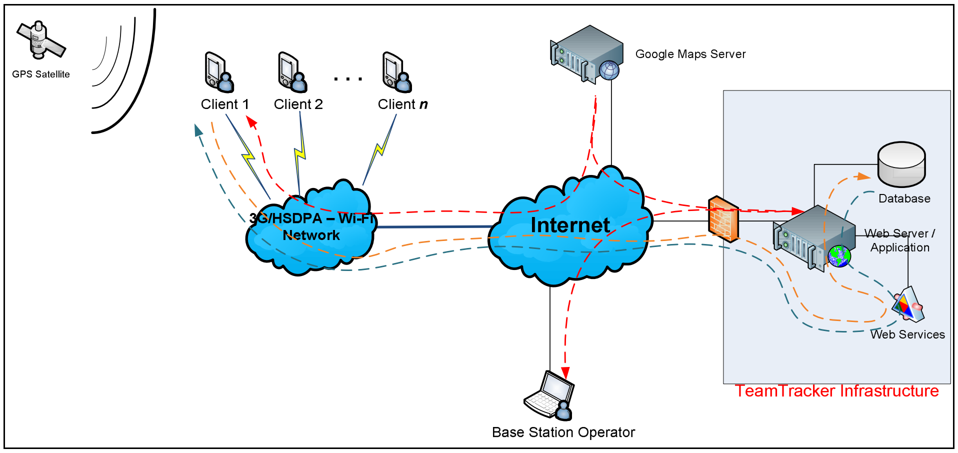


Figure 4. “TeamTracker” System Architecture

“TeamTracker” system provides clients the ability to connect to a web server. Once the clients are authenticated, by utilizing web services, they may update their current position using GPS derived longitude and latitude parameters. This data is stored on the web server’s database in real-time and is viewable at the base station. Another function of the application is to allow users to collaborate with other online clients by posting and receiving dashboard updates which allows each user to view other user’s whereabouts as shown in Figure 7.

The base-station type scenario which is also provided by the system allows administrators to simultaneously track multiple connected clients, or individually. The administrator, or base-station operator, also has the ability to view past or current tracking/position data per user that enables them to visually view and utilize path analysis.

7.1 Client Application

The client application consists of a configurable interface which allows users to login to the TeamTracker web application and consume its web services. In order for a user to be able to use the service, the user must firstly login, as illustrated in Figure 5. Client application also consists of global settings that allow users to set certain parameters that are explained in Figure 6.


```
public void Authenticate()
{
        if (username == String.Empty || password == String.Empty)
        throw new Exception("Username/Password not set!");
        try
        {
                serverResponse(service.authUser(this.username,
                                                this.password));
        }
        catch(Exception ex)
        {
                if (OnServerResponse != null)
                OnServerResponse("Unable to authenticate
                                    with the server.", new
                                    ResponseEventArgs(Response.COMMUNI
                                    CATION_ERROR));
        }
}
```

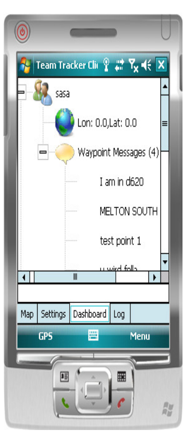
Figure 5. "TeamTracker" Client Authentication Procedure

The client application allows certain parametric settings to be configured. These include:

- Ability to track to server or XML file
- Ability to set the GPS to server polling interval – this enables clients to adjust polling cycle in applications where position update is not needed every second, but rather every 60 seconds.
- Functionality to record tracking – Clients may record their tracking which is logged back to the web server database through web services.

Figure 6. "TeamTracker" Client Application Settings

7.2 Dashboard



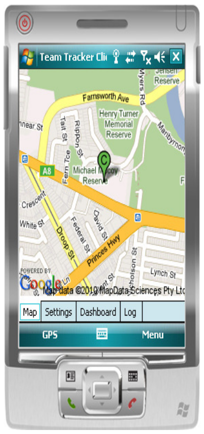
Dashboard enables users to communicate their positions and share messages. Each user may post messages that are stamped with their current position. The data is fed into the database and distributed to the base station and other individual users.

Dashboard messages act as application waypoints allowing users to navigate to dashboard positions and check other user positions.

Figure 7. "TeamTracker" Dashboard Update Feature

7.3 Google Map API

Utilizing Google Static Maps, a web-request can be constructed to Google Maps server by providing set of parameters as a URL and call the builder construct which will initiate the web request and receive the map response.




```
public string MapUrl
{
        get
        {
                return String.Format(CultureInfo.InvariantCulture,
                        "http://maps.google.com/staticmap?center={0},{1
                        }&size={5}x{6}&markers={0},{1},greenc&zoo
                        m={2}&maptype={3}&key={4}",_position.Latitude,
                        _position.Longatude, _zoomLevel,_mapType,
                        _apiKey, _xSize, _ySize);
        }
}

builder.Position = pos;
builder.ZoomLevel = 15;
builder.YSize = picMap.ClientRectangle.Width;
builder.XSize = picMap.ClientRectangle.Height;
builder.GoogleMapsAPIKey = GOOGLE_API_KEY;

LocationMap map = new LocationMap(builder.MapUrl);
picMap.Image = map.Map;
```

Figure 8. "TeamTracker" utilizing Google Maps API locally to the device

The client application is also responsible for obtaining GPS data. The data is retrieved by utilization of Windows Mobile 6.0 SDK API. The API enables connectivity to the GPS device and asynchronous callbacks which provide GPS position data and GPS device data. Below class diagram illustrates the GPS API class model.

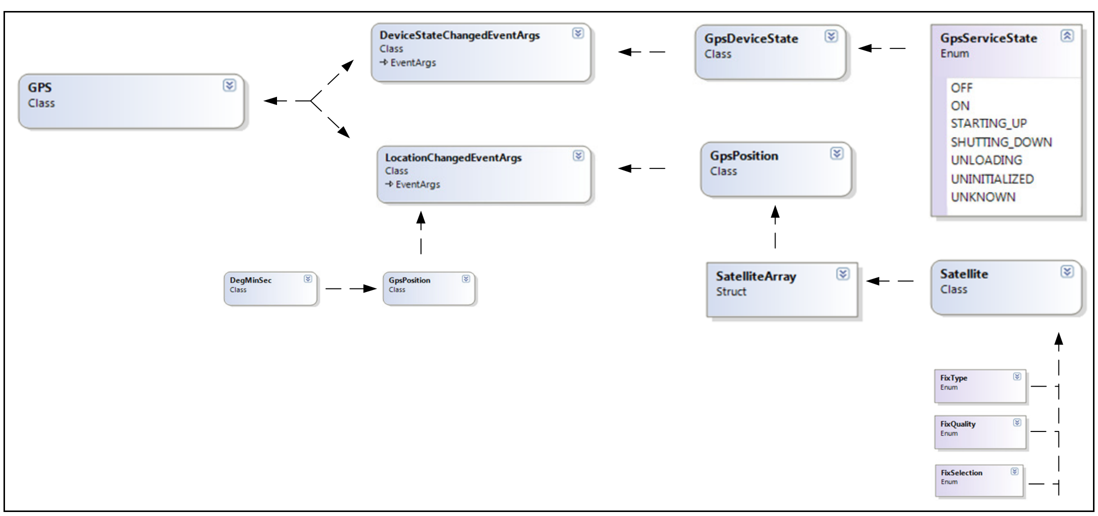


Figure 9. GPS Managed API Wrapper

When class is consumed by the application, it allows easy utilization of class methods for establishing connection with the device and invoking GPS related events. Below ***code*** demonstrates GPS device connectivity and asynchronous event invoking. In the example below, an event handler is added on **LocationChanged** and listen to GPS position changes.

```
private Gps gps = new Gps();

gps.Open();
gps.LocationChanged += new LocationChangedEventHandler(gps_LocationChanged);

void gps_LocationChanged(object sender, LocationChangedEventArgs args)
{
        if(this.InvokeRequired)
       {
              this.Invoke(new gps_LocationChangedDelegate(gps_LocationChanged),
                             new object[] { sender, args });
       }
       else {
        position.Latitude = args.Position.Latitude.ToString();
        position.Longatude = args.Position.Longitude.ToString();
      }
}
```

Base-station component of TeamTracker system consists of a web application allowing base-station operators or administrators to manage user accounts, but more importantly ubiquitously monitor all users in real-time. This is accomplished seamlessly by implementation of timed AJAX request queries to a backend database. The data returned is in form of JSON (JavaScript Object Notation) and then evaluated, which in turn creates a usable object. By enumerating through the object data can be obtained to be used in rendering map or page details.

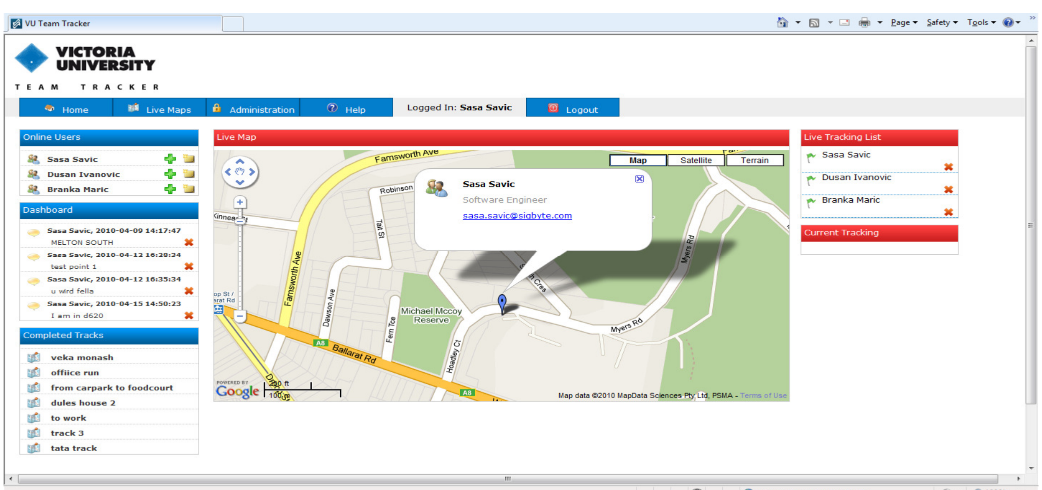


Figure 10. Multiple user tracking

The map provider of choice was Google Maps. The Google Maps API provides a comprehensive and feature rich model that is implemented in JavaScript and is cross-compatible with all browsers. Figure 10 illustrates application's capability to be monitoring multiple users at any given time displaying their latest position in near-real-time (The data that is being rendered on the screen occurs asynchronously at 3 – 5 second intervals). Another feature of the application is the ability to be displaying user dashboard updates and marking them against their relevant position.

Track recording capability implemented in the application allows base-station operators to plot movement of users and use them as statistical comparison for path analysis. Each longitude and latitude interval point is recorded in the database and can be consumed by further extensions of the application to produce numerous statistical analyses.

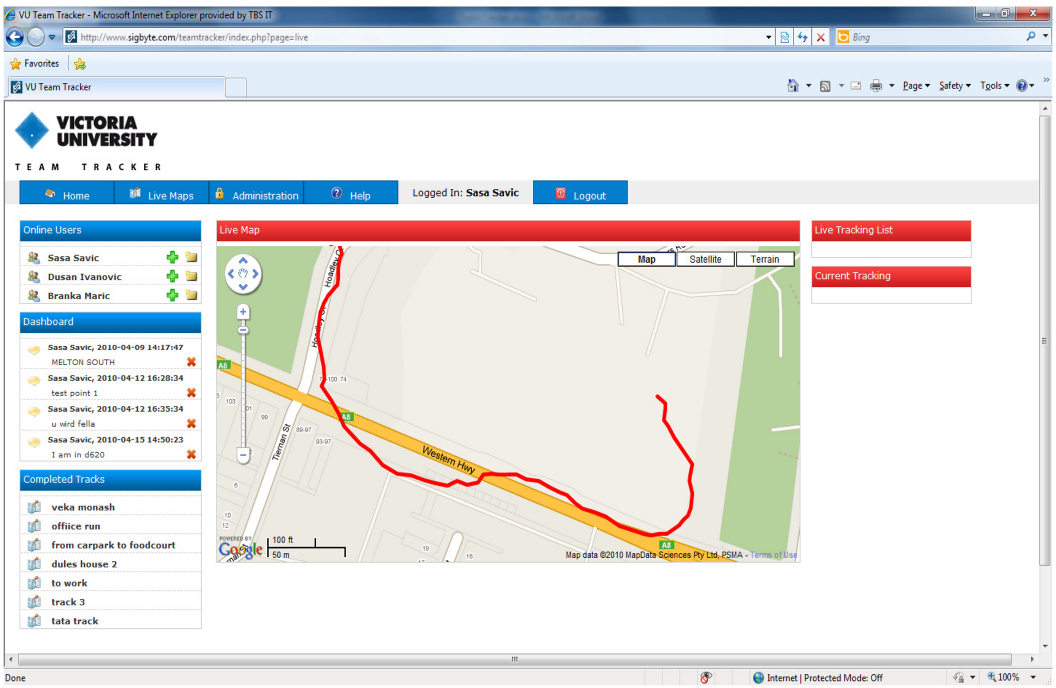


Figure 11. Path tracking analysis

Illustrated in Figure 11 is a map which provides tracking path. When tracking multiple paths, the data becomes transparent to the operator, providing them with point position per user. Data mining can further be applied to the model, which could inherit different type of analysis on the position data.

7.4 Database Model

Relational database model fits the TeamTracker design in such way that it provides web service and web application compatibility. The model is scaled across all services and components. WSDL endpoint services, which are endpoint function definitions map back to the data model defined in database. Furthermore, the model is then defined as abstracted functions that provide transparent data access to the various components of the system, such as; client side application, web application and web services which are defined in the next section, as shown in Figure 12.

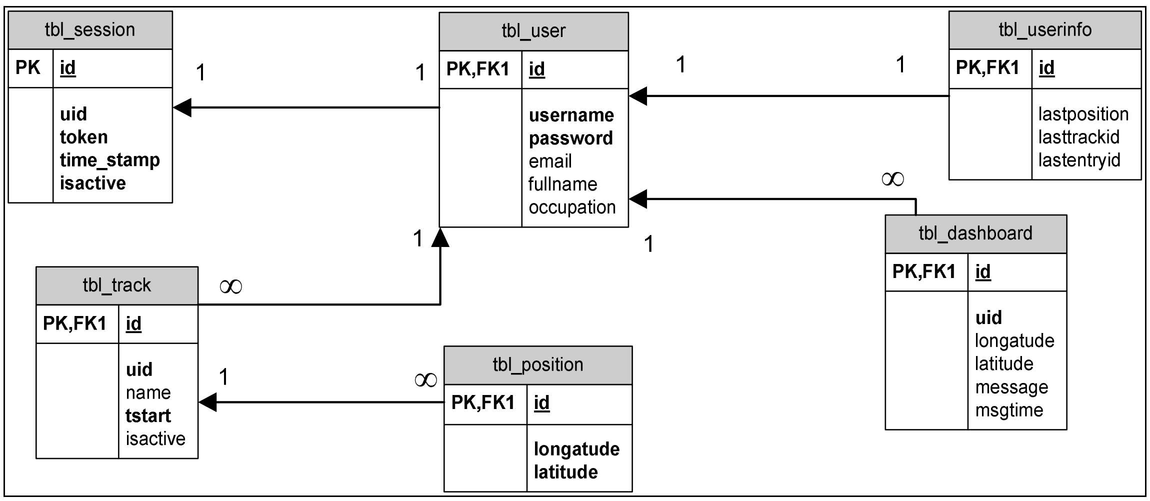


Figure 12. TeamTracker Database Model

7.5 Web Services

Core component of TeamTracker are the web services defined by the WSDL definition as shown in Figure 13. In the current model, there are eight operations (functions). Below is the detailed description of **authUser** function, the rest can be viewed at http://www.sigbyte.com/teamtracker/services/tracking.php and the web services definition can be viewed by adding "**?WSDL**" to the end of the URL:

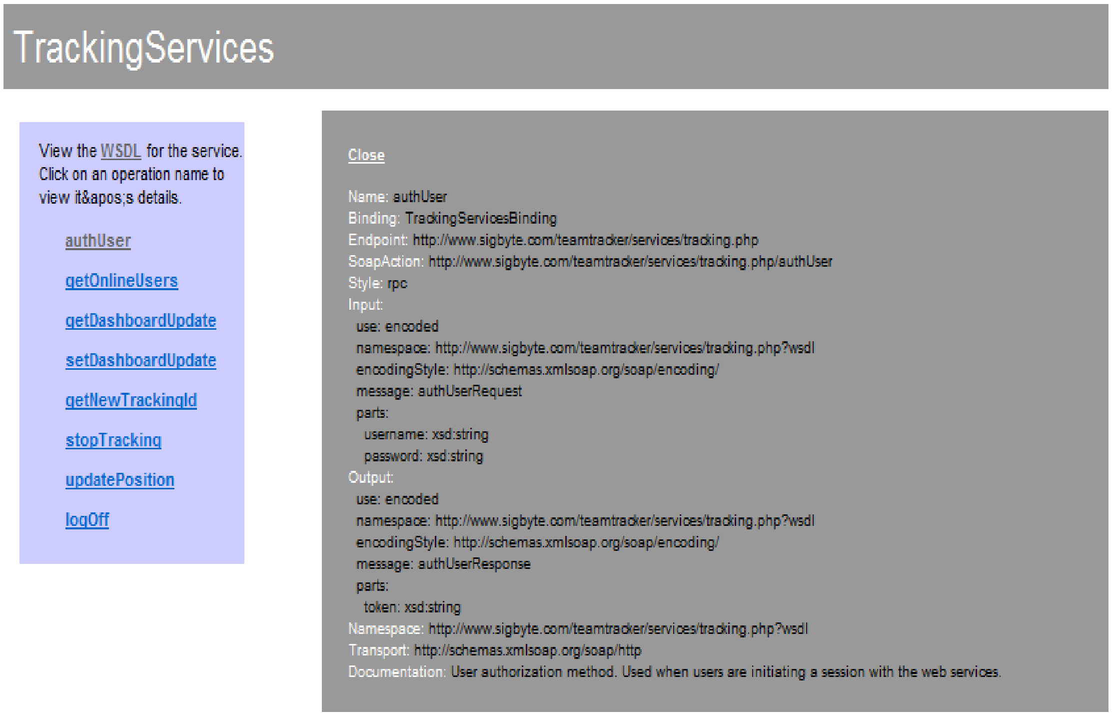


Figure 13. Web Service functions

Further looking into the **authUser** operation, a detailed code definition is defined below.

```
$webServer = new soap_server();

$namespace = 'http://www.sigbyte.com/teamtracker/services/tracking.php?wsdl';
$webServer->configureWSDL('TrackingServices');
$webServer->wsdl->schemaTargetNamespace = $namespace;

$methodName = 'authUser';
$input = array('username' => 'xsd:string','password' => 'xsd:string');
$output = array('token' => 'xsd:string');
$soapAction = false;
$style = 'rpc';
$use = 'encoded';
$description =  'User authorization method. Used when users are initiating a session with
the web services.';
$webServer->register($methodName, $input, $output, $namespace, $soapAction, $style,
$use, $description);
```

Figure 14. Web Service functions

8. CONCLUSION AND FUTURE WORK

Based on the current framework, "TeamTracker" has been created successfully from design and development to implementation. Multiple users are able to login to the services and provide the server with their position data. The system is able to track user positions and record their movements based on adjustable time intervals. The base-station operators or administrators are able to view users and their data asynchronously and able to track them in real-time by utilization of Google Maps and the API. This open ended architecture also provides other developers with the platform ability to consume the web services and integrate their own devices. The application is not limited to mobile phone devices.

The abstract design of the system is proved to be positive and easily extendible. "TeamTracker" can further be extended to include user groups and implementation of rule based tracking. The system could also be further enhanced to provide administration ability to adjust certain application parameters. This would include the data polling interval and maps to be rendered with greater dynamic abstraction from the data queried from Google API servers. The project exhibits enormous potential for use in search and rescue, fleet tracking, general team collaboration or agricultural analysis.

REFERENCES

[1] MSDN, Windows Mobile Documentation, viewed 19th April, 2010, < http://msdn.microsoft.com/en-us/library/bb158532.aspx >

[2] MSDN, Introducing Visual Studio, viewed 18th April, 2010, < http://msdn.microsoft.com/en-us/library/fx6bk1f4(v=VS.100).aspx >

[3] .Net Compact Framework Architecture, viewed 21st April, 2010, < http://msdn.microsoft.com/en-us/library/9s7k7ce5 >

[4] Web Services Description Language, viewed 22nd April, 2010, < http://www.w3.org/TR/wsd >

[5] Isakow, A. and H. Shi, "Design and Development of GlobeTrotter - An Innovative Location-Based System using J2ME ", International Journal of Computer Science and Network Security, Vol. 8 No. 3 pp. 201-207.

[6] Ranjan R. and H. Shi "GLOBETRACKER – a Wireless Tracking System", International journal of Computer Networks & Communications, Vol.1, No.3, October 2009, pp. 125-130.

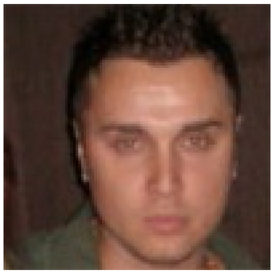
[7] 8 Motions Server, viewed at 26th April 2010, < http://corp.8motions.com/ >

[8] Ortiz, 2007, A Survey of Java ME Today, Sun Developer Network (SDN), viewed 21st August, 2010 <http://developers.sun.com/mobility/getstart/articles/survey/>

[9] K. Fertalj and M. Horvat, Comparing architectures of mobile applications, viewed 22nd August, 2010, arxiv < http://arxiv.org/ftp/cs/papers/0703/0703041.pdf >

[10] K. Victor, Y Vandewoude and. Barbers, Application platforms for embedded systems: Suitability of J2ME and .NET Compact Framework, viewed on viewed 22nd August, 201

Authors



Sasa Savic is a minor thesis research student at Victoria University of Technology, Melbourne. He obtained a Advanced Diploma of Computer Systems Engineering in 2002 from Royal Melbourne Institute of Technology (RMIT). In January 2006 he received a Bachelor degree in Computer Science from Victoria University of Technology. He currently works at Telstra, a core telecommunications company, as a software engineer. He can be reached on sasa.savic@live.vu.edu.au

Dr. Hao Shi is an Associate Professor in School of Computer Science and Mathematics at Victoria University, Australia. She completed her PhD in the area of Computer Engineering at University of Wollongong and obtained her Bachelor of Engineering degree at Shanghai Jiao Tong University, China. She has been actively engaged in R&D and external consultancy activities. Her research interests include p2p Network, Location-Based Services, Web Services, Computer/Robotics Vision, Visual Communications, Internet and Multimedia Technologies.