Live Tracking Framework  
for Public Transportation Systems

László Barabás*, Réka-Andrea Károly*, Károly Simon*

* Babeș-Bolyai University, Cluj-Napoca, Romania
Faculty of Mathematics and Computer Science,  
Computer Science Department,  
barabas.laszlo@evoline.ro  
karoly.reka@evoline.ro  
ksimon@cs.ubbcluj.ro

Abstract—Live tracking is essential in the case of public transportation systems. Furthermore, making public the locations of public transportation units can also be useful. It not only helps the transportation providers, but the clients, as well.

A general live tracking framework is proposed. It is applicable for any kind of logistics system. In this study the Romanian railway system is used as a test environment. Train movements are simulated based on timetables. Using the framework a web application is developed. A map service is used for displaying the railway system and the simulation.

Keywords: live tracking; traffic monitoring; public transportation; web application; Romanian railways; simulation

I. INTRODUCTION

When traveling by train in Romania there are several websites where users can find information about the timetables (http://www.infofer.ro/, http://www.mersultrenurilor.ro/, http://merstrenuri.ro/ etc.). However, sometimes these sources give different data. It can be frustrating when a person is in a hurry and cannot decide which page was last updated, where the actual scheduling can be found.

With real-time train tracking, the system developed using the proposed framework ensures that the checked timetable is the actual one. Users can also be notified about possible delays.

By providing information of public interest, the application has a large target audience. It can be easily accessed by a large variety of devices.

The project offers a better user experience providing a map-based interface, supporting zooming and dragging operations. The moving trains are visualized on the map, and there is no need for browsing complex timetables. Users can see the position of the train even between two stations, which information cannot be displayed using simple timetables. The application provides possibility for searching and filtering trains.

Having the proper architecture, guaranteed by several design patterns, the application can be easily changed and any other logistic system can be considered. The only modification that needs to be done is on the level of the used data source. The data source provides information about the tracked vehicles and the transportation network (routes on which the vehicles are moving).

For testing purposes a timetable-based simulator is used for providing information about the current location of the transportation units. The simulator can be easily replaced by a subsystem, which provides real GPS data. In this way the exact location of the transportation units can be visualized. A special method is used for making proper corrections if the received coordinates are not exactly fitting to the transportation network (e.g. inaccuracy of the GPS device).

Redundancy can be introduced in the system. The GPS-based tracking module can be combined with the simulator. If there is no received data from the GPS device (e.g. the transportation unit is in a region, which is not covered by any communication network), the location of the unit can be approximated using the simulator. Artificial Intelligence methods can be used for providing a better position approximation, and for making delay predictions.

In section 2 the used technologies are briefly described. In section 3 the framework architecture is presented. The application developed using the framework is presented in section 4. In section 5 there are several similar applications mentioned among the most important differences regarding ours. The last section describes further development possibilities.

II. USED TECHNOLOGIES

The framework is based on a client-server model, having a service-oriented architecture. On server side mainly Microsoft technologies are used. Some JavaScript APIs and libraries are used on the client side. These technologies are briefly presented in the following subsections.

A. ASP.NET MVC 3

The ASP.NET MVC3 framework [1] is a Microsoft product available from the 4.0 version of the ASP.NET. It was published on the 13th January 2011 as the successor of ASP.NET MVC 1 and 2. ASP.NET MVC 3 is built on the previous versions, it provides all the functionalities provided by these versions, but also adds great features for simplifying the code and allowing more extensibility.
The framework offers a new type of project called ASP.NET MVC 3 Web Application. This kind of project template with its directory structure and process flow obligates developers to use the MVC architectural pattern in their web applications.

The framework provides support for rendering web pages in HTML5. Since the scope of the HTML5 is to standardize the HTML across browsers and devices, the ASP.NET MVC3 together with HTML5 increases the usability and interoperability of our framework.

This technology is used for publishing and displaying the maps and the tracked objects on any kind of device that has internet browsing possibility.

B. WCF – Windows Communication Foundation

WCF [2] is a framework which provides a unified programming model for rapidly building service-oriented enterprise applications, which are communicating across networks. This technology was built upon the .NET framework as a separate package, but from the 4.0 version of .NET it is an included part. Interoperability, flexibility and optimized WCF-WCF communication are important features provided by the framework. Another aspect is that WCF services can be easily configured, different bindings (communication protocol) can be changed quickly without the need of making any modifications in the source code.

Our framework is a multi-tiered one. There is a layer for the data services and another one for the web application. The communication between them is realized using several WCF services.

C. JavaScript

JavaScript is an object-based scripting language, which can be interpreted by web browsers. We used JavaScript for making our web application dynamic and for displaying maps and tracked vehicles with the help of third-party JavaScript libraries. These libraries are the: Google Maps and Open Layers.

D. Google Maps API

The Google Maps API [3] is used in the project for map displaying and rendering. It also gives the possibility to add zooming and navigation controls, upper layers and custom icons to the map.

E. OpenLayers API

It has almost the same functionality as the Google Maps API. Among others an important difference between them is the licensing. While the Open Layers API [4] is an open source framework, some of the Google Maps API features can be used only with restrictions.

In both cases different layers are used for displaying the map, the possible routes of the tracked vehicles and the tracked objects.

F. jQuery

The different browsers expose the same functionality via different JavaScript functions. jQuery is a cross-browser JavaScript library, which has the purpose of standardizing the way of accessing the above mentioned problematic functionalities. More precisely it hides them under a common API.

In our framework it is used to display the search and filter menu and to animate the pop-ups that appear when the mouse pointer is over a vehicle (train) icon.

III. ARCHITECTURE

The framework is composed by several subsystems. The architecture of the system is briefly presented in the following subsections.

A. Main Components

Generalization plays the main role in the system architecture. The monitorized logistics system can be easily changed, the components are well separated, and in this way elements can be replaced with minor modifications. Using the appropriate multitier architecture, the layers can be extended easily by adding new features.

The application has six main components, see “Fig. 1”:
- Web Server
- Shape Data Server
- Train Data Server
- Train Movement Simulator
- Browser
- Map related API and database

The shape data server stores and exposes information about the transportation network. The train data server receives, stores and serves information about the transportation units. The current location of these units is provided by the train movement simulator. In case of real time tracking this is changed or combined with a subsystem providing GPS-based localization.

The above mentioned components do not need to be on the same physical or virtual machine. Each part has its own lifecycle and can work separately. The communication between them is based on WCF services. Data transfer can be done in different ways. The web

![Framework architecture](image)
server communicates with the two data servers using TCP/IP binding, but in the case of the simulator a more general binding is used, called WS HTTP Binding. This kind of binding realizes SOAP web service, which provides interoperability with other platforms that support web services. This offers the possibility for replacing the simulator with any server providing real GPS data that supports web services.

B. Map and transportation system visualization

There are several JavaScript libraries for rendering GIS maps. Our purpose was to give possibility to use any of them in our framework for displaying the information. For this an abstract JavaScript layer was created, which could wrap any existing map rendering library. Our implementation includes two major map rendering libraries: the Google Maps API and the OpenLayers API.

“Fig. 2” illustrates the architecture of the map displaying subsystem. For testing purposes a shape file related to the Romanian railway system is loaded.

The subsystem contains seven main components:
- Google Maps API and Database
- Shape File
- Shape Data Server
- WCF Shape Service
- WCF Shape Client
- Web Server
- Browser

The shape file contains the coordinates specifying the possible routes of the tracked objects. This data is processed by the Shape Data Server. Any changes made to the file will be observed by the server and future requests will receive updated data. While processing client requests the web server queries the shape information from the Shape Data Server and sends it to the corresponding client’s browser.

The communication between the Shape Data Server and the web server is based on the Microsoft’s Windows Communication Foundation technology. The implementation of the WCF based communication between Shape Data Server and web server is included in the WCF Shape Service and WCF Shape Client components accordingly. In this case NetTcpBinding (uses binary encoding and it is the fastest way of message exchanging between endpoints) is used because of the large amount of transferred data.

Google Maps API is used for displaying and rendering the map and visualizing the network as an upper layer. The same applies when using the OpenLayers API.

C. Display and update mechanism for vehicle positions

The transportation units (trains) are illustrated with icons on the map. The architecture of the visualization subsystem is depicted in “Fig. 3”.

WCF technology is used for the communication between the web server and the train data server.

Since the Romanian trains are not equipped yet with GPS devices, a train movement simulator has been implemented for generating train positions. This simulator is based on the Romanian Railway Company’s public timetable.

The Train Data Server constantly receives information regarding the tracked objects’ positions and states. These data are stored locally. The update of an object’s position is done when the received data differs from the already stored one. Another piece of information that is received by the Train Data Server is the train’s state: whether it reached its destination or not. One minute after a vehicle reaches its destination it should be removed from the map.

Each web client refreshes periodically the positions of the displayed vehicles. In the background a HTTP request is sent to the web server for querying updated information about the tracked objects. The web server in order to prepare and send back the desired data has to forward the
query to the Train Data Server. This will provide the necessary information. In order to reduce the size of data packages the response of an update request will contain only those vehicles that were refreshed since the last request was received from the corresponding client.

IV. FUNCTIONALITY

For using the application a page request has to be sent by the users to the web server using a web browser. After accessing the appropriate link a map will appear with three layers, see “Fig. 4”:

- Geographical map (Romania)
- Transportation network (railways)
- Icons representing the vehicles (trains).

In the case of the presented application three types of icons are shown on the map: red circles representing interregional trains, green icons for intercity and gray icons for regional trains.

The navigation and zooming control bars are displayed on the left side. By clicking on the arrows, users can navigate on the map. By adjusting the zooming control bar, the dimension of the map can be increased or decreased. The navigation and zooming can also be done using the mouse. By holding down the left mouse button and moving the mouse users can navigate on the map. With scrolling up or down users can zoom in or zoom out.

A search and filter menu is displayed on the upper right corner of the page. This can be hidden by clicking on the “Hide Search-Filter” button. Searching can be done by using the id (name) of the vehicles (trains). Auto complete feature is also available. After selecting a vehicle (train) from the list, the map is automatically centered and zoomed to the chosen unit (train).

In the lower right corner of the page there is a button, which enables switching between Google Map and Open Street map.

V. SIMILAR APPLICATIONS

There are similar applications like Flightradar24 (http://flightradar24.com) or Swisstrains (http://swisstrains.ch). The first one is a monitoring application for airplanes around the world equipped with special tracking devices. The Swisstrains application shows the trains’ movement in Switzerland.

Both applications mentioned above uses Google Maps API. Our application’s scope was to be flexible, being able to use any kind of map rendering API. Furthermore, our framework is a general one from the tracked objects’ point of view. It can be set up for monitoring any kind of logistic system.
VI. CONCLUSIONS AND FURTHER DEVELOPMENT

A general framework for live tracking and traffic monitoring has been developed. The transportation network and the position of the transportation units are displayed using a web-based user interface. The system can be used for any type of logistics system.

The application has been tested for the Romanian railway system. A timetable-based simulator software has been used for approximating the current position of the transportation units (trains). The result is displayed on a map. Navigation, zooming and searching operations are supported.

The project is still a work in progress, so new features are planned to be added.

Another functionality would be to enable route searching. In this case users could choose a departure station and a destination and the application would suggest different possibilities.

As a next step the application should be tested with real GPS data. Based on these data and the official scheduling delays could be calculated.

After obtaining real data from the trains (or any other logistics system that the framework uses) statistics could be calculated for a better approximation of the vehicles’ positions between two checkpoints.

REFERENCES