Real-Time Tracking Management System

Jose C. Almeida[1] and Artur M. Arsenio[2(✉)]

[1] Instituto Superior Tecnico/Lisbon University, Lisbon, Portugal
[2] Universidade da Beira Interior & IST-ID, Covilhã, Portugal
arsenio@alum.mit.edu

Abstract. Over the latest years many commercial Real-Time Tracking management systems (RTMS) were introduced into the market. The solutions started to be designed for single tracking purposes, but vendors soon realized that valuable tracking benefits would result by the appropriate design of a service layer. From the many RTMS solutions currently on market, it was not found a specific or suitable solution for general Law Enforcement Agencies (LEAs). A list of requirements was gathered from one of the Portuguese LEAs, GNR (Guarda Nacional Republicana) to fully understand the generic daily challenges. Main requirements raised by GNR strive with that address issues like cost, data exchange security, bi-directional communication services, performance, network fault tolerance and user-friendly interfaces. This paper presents a RTMS solution and a service layer specifically conceived for LEAs. It will be shown that learning over risk assessment mappings may bring additional benefits to LEAs. Experimental tests were performed to quantitatively measure the solution behavior.

Keywords: Real-Time tracking management system · Law enforcement agencies · Location based services · Security · Mobile

1 Introduction

Law enforcement agencies (LEAs) have primary responsibility for the maintenance of public order, prevention and detection of crimes. They fight on a daily basis for the indispensable order of society, but incidents are becoming day-after-day more complex, and LEAs human resources are limited. Therefore, it is required to maximize the efficiency of these organizations. Good coordination and support levels requires high availability of information on a real-time basis, not only directly to the field workforce but also for the ones in charge of the corresponding decision support. Furthermore, it is necessary to provide services to the workforce, such as monitoring agents' health status and providing them rapid support in medical urgencies.

Informal meetings with a Portuguese LEA' officers were held to understand the operational and management challenges of an LEA. The most relevant challenge is the use of radio (voice) as single tool of communication between agents. Despite radio network coverage and voice quality have been improved over the last 5 years, it was concluded that the single use of radio is far to fulfill the level of information on real-time basis nowadays required. From a management perspective, there is a significant lack of efficiency to get a clear (mind) picture of the fleet location using a single

B. Mandler et al. (Eds.): IoT 360° 2015, Part II, LNICST 170, pp. 475–483, 2016.
DOI: 10.1007/978-3-319-47075-7_51

voice channel (radio). This issue leads to the adoption of a Real-Time Tracking Management System (RTMS), a system that provides the location of the fleet against a geographical map on a real time basis. The inefficient process of enquiry the fleet location, vehicle by vehicle, using voice, would in contrast be replaced by a system that provides instantaneously a clear picture of the fleet location. However, the location information is not the only issue raised.

The adoption of an RTMS opens the doors for delivery of a new level of services. This service layer enables for instance a fleet to receive a job/mission with specific geographic coordinates and respective navigation support to easily and time efficiently get on site, or the issuing of an SOS signal with respective coordinates requesting assistance from nearby agents or for medical support. Furthermore, it enables learning common patterns of risks by geographical areas according to past history.

Many RTMS solutions were designed to meet large market segments like transport of goods, public transportation, or utilities (e.g. big fleet workforce spread by the territory) [1]. This paper proposes a simple, efficient, secure, low cost solution. It is open to the integration of new specific LEA services and features, which can distinctively enhance the LEAs activity management and consequent citizen's safety.

2 State of the Art

A RTMS is a system capable to monitor in real-time an asset motion and corresponding valuable data (e.g. temperature, average speed, fuel consumption, etc.) This system is fed by a timely ordered sequence of data, fetched from many possible data sources (e.g. GPS receiver for geospatial information purposes). RTMSs can have a bi-directional communication, enabling the transmission of data/control commands from/to data sources, respectively. RTMSs manage multiple streams of data originated by those sources, and display it to the end-user using a suitable representational model (e.g. geospatial data against a geographical map). Despite the wide usage scenarios for RTMSs, they are mainly composed by 3 major components:

(a) Tracking unit (data source), which captures the location information at specific time intervals, and reports (push/pull) it to a tracking server. Location information may be coupled with data from other sensors/modules. In case of bi-directional communication, the tracking unit has the (additional) responsibility of receiving data/control commands from the tracking server. Smartphones, as well as other daily-life gadgets (e.g. tablets with GPS built-in), can be easily turned into a tracking unit device, simply running a GPS tracking software. This is suitable for human related tracking applications [2].
(b) Tracking server: It has 3 responsibilities: receive data from the tracking unit, securely store it, and serve this information on demand to the user front-end. In case of bi-directional communication, tracking server has the (additional) responsibility to transmit data/control commands to tracking units.
(c) RTMS Front-end: The user interface determines how to access information, or view asset valuable information such as location. In case of bi-directional communication, end-user might use front-end interface to transmit data/control commands to tracking units (through tracking server).

Communication between tracking units and tracking server is critical: the lost or even delay of data over the network might turn impossible to use an RTMS. For instance, the use of LEA RTMS on a criminal car chase requires an efficient communication system to provide details of the fleet location (along with other information) in near real time. Otherwise, LEA criminal intersection plan implementation might not succeed due to the latency/disparity of the fleet location on the RTMS (LEA decision point) and real location on the field.

Since LEAs exchanged data may contain valuable information to criminals, such as the real-time location of LEA workforce, data encryption plays an important role. The power consumption of encryption operations (computationally expensive) must be taken into account, since tracking units have a limited energy source.

Muruganandham [3] proposed Real Time Web based Vehicle Tracking using GPS. Tracking units report over GSM the vehicle information to a tracking server, storing reports in a database. They followed the classic (most used) approach of design/build a customized unit, in contrast to our approach of using a Mobile market device (Tablet or Smartphone) with all built-in. The design of a customized unit allows for better performance and efficiency. However it has disadvantages such as the effort/time cost to prototype a hardware solution, and the final price. Some requirements identified by the authors, such as the information of the vehicle ignition (on/off) and door status (open/closed), adds additional complexity to the Mobile device solution. Nevertheless, it could be solved with a simple CAN-BUS to Bluetooth device, assuming CAN-BUS is deployed in the vehicle.

Hasan et al. [4] have proposed a cost effective method of object tracking using GPS and GPRS technology. The user can view the present location of the object as well as the past history of its movement using Google Map and Internet. They take advantage of the IMEI number uniqueness to be used on the authentication of the device. Hence, the Tracking server drops all the messages that do not have a valid IMEI stored in the database. This is a simple security approach that resides in the secrecy of the IMEI, which can be easily exploited by man in the middle attacks and spoofing. Another important characteristic of this work is the use of HTTP POST method to transmit data between the tracking units and server. In that way, the implementation (e.g. validation, data manipulation, etc.) of the tracking server is developed in PHP, which is not the best solution in terms of performance. This choice represents a high cost of CPU and memory per vehicle. A shared tracking service business model shall be definitively more cost-efficient than the tricky shared web hosting proposed. Shared tracking services follow the growth of cloud computing usage and optimized elasticity of resources (e.g. GPSgate and GPS-Trace).

3 Solution

The overall solution is composed by a mobile app running on a tablet in each vehicle, and a tracking server along with a database and web server. The vehicles share a secret key with the tracking server, and exchange data securely through mobile operator network and (eventual) third party networks. The Central station(s) manage the fleet through a web based solution, as shown in Fig. 1.

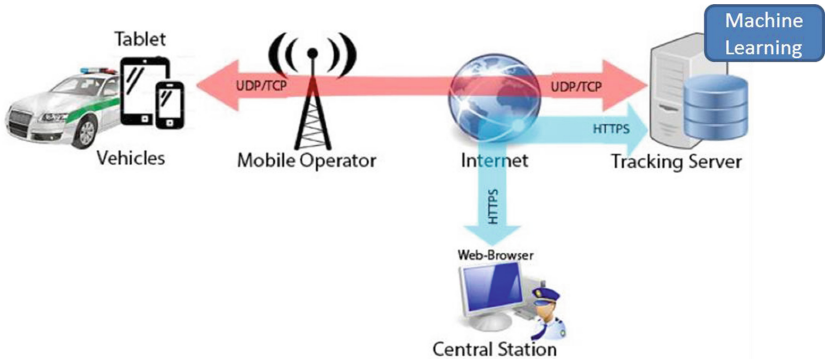

Fig. 1. High-level solution architecture.

Our solution strive to enhance the LEAs daily operational requirements through offering the following features:

- **Real-Time Monitoring:** Central Station can monitor fleet status (e.g. location) in real time (up to the second).
- **Alarm:** Vehicles can trigger an SOS alarm in Central Station. Central Station can send multiple POI to Vehicles.
- **Job Dispatcher:** Central Station can dispatch a job containing specific location to vehicles.
- **File exchange:** File transmission between vehicles and Central-Station.
- **Report:** Vehicles can report geo-referenced events.
- **Photo:** Vehicles can take photo directly to Central Station.
- **LEA data:** Vehicle terminal have local access to synchronized list of Person (suspects and lost) and Vehicles (suspects and robbed).
- **Risk Map:** Vehicles can easily see a risk representation over the map (layer). The layer is dynamic; it changes accordingly to the date and time of the day.
- **History:** All information is stored on the database, so the LEA is capable of representing previous fleet locations, and corresponding events, on map.
- **Security:** All data exchanged is confidential, authenticated and incorruptible.
- **Network:** Tolerant to networks faults through retransmission of data lost.

3.1 Central Station (Tracking Server)

The tracking server is coded in Java for high code portability, since existing technological infrastructures may widely differ between LEAs. The server architecture solution is designed according to Fig. 2-a.

DB Manager: Module responsible for all operations in the server database.

File Manager: Module responsible for all the local file system operations. In addition, features a simple versioning of files, enabling proper file synchronization.

Operation Handler: Module responsible to perform operations requested by the vehicles applications.

Application Manager: First module loaded by the server. It is responsible to load all the required data at start (e.g. server configuration). User credentials are also loaded, and it is kept on memory, since it will be accessed very often (e.g. anytime a message arrives). Once all data required by the server is loaded, Application Manager loads the Operation Handler module. Nevertheless, Application Manager remains on active cycle collecting and storing (via DB Manager) periodically all RTMS relevant data. The data storage is prioritized by the period of verification and thread priority. For instance, real-time status messages are updated on database with minimum delay comparing to the status history update (checked every second, and only stores 1 value every 10 s, using the minimum priority thread). It is responsible for management of allocated resources that each vehicle app demands from server.

Security: Module responsible for the encryption, decryption and validation of data (e.g. anytime a message arrives) using the open-source "Bouncy Castle" lightweight cryptography library on Android OS.

Message Handler: Module responsible to receive live status messages from vehicles apps, and to send control messages to the corresponding vehicle app (e.g. Dispatch a job). The communication is preferably done through UDP, but TCP is also supported. Both work in concurrency on the server and the logic of decision is implemented on the client side (mobile app).

The Database (depicted on Fig. 2-b) was designed for easy management. No security measures are applied by the tracking server solution to the database content, since database encryption is a feature supported natively by most popular DBMSs.

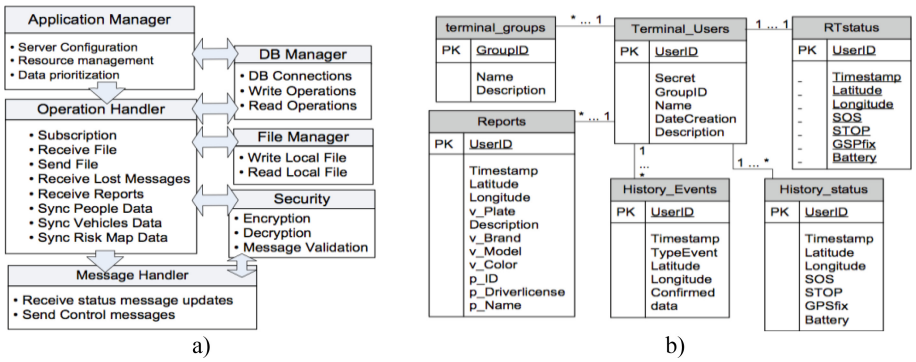

Fig. 2. (a) High-level architecture of tracking server. (b) UML Database model.

3.2 Vehicle

The mobile app is coded in Java for Android (API 17) and can also run on other OS through an emulator (e.g. Bluestacks). The overall design is shown in Fig. 3-a.

a) b)

Fig. 3. (a) High-level architecture of mobile app; (b) Representation of risk map feature.

Operation Handler: Module responsible to perform operations. All the operations start with a TCP socket connection originated by the mobile app to the tracking server (e.g. File transmission).

Application Manager: Module responsible for all the user interface of the mobile app. It enables the user to trigger events through the mobile app (e.g. SOS alarm, Report), visualize current location and LAE data (e.g. missing persons). The first operation performed by this module is the subscription.

Message Handler: Module responsible to fetch data from devices modules (e.g. GPS), and send it as status messages updates to the tracking server (each second). The module also receives control messages, which are commands originated by the tracking server (e.g. Job dispatch). The communication is preferably done through UDP, but TCP is also supported (when the device is behind a NAT/firewall [5]). The detection of NAT/Firewall happens when a local IP is assigned to the network device. If the assigned IP is no longer local, the system automatically switches to UDP.

The mobile app user interface was designed to be simple and intuitive: contains only 3 buttons on the dashboard. An "Auto View" automatically centers the vehicle position in the screen. A high accessible SOS button triggers an immediate support request at the Central Station. A Risk Map feature (implemented on Google Maps Android APIv26) provides an over-layer (the risk) on the map, as shown in Fig. 3-b.

3.3 Machine Learning

Events reported by the vehicles, with their correspondent localization, were employed to build a dynamic risk map layer, which changes accordingly to the date and time of the day. Appropriate learning of criminal patterns may allow the exploitation of this feature with important data, from criminal investigation analysis perspective to crime prevention. Indeed, current work is exploiting two learning techniques to gather more information from risk maps: back-propagation neural networks, a supervised learning approach, and an unsupervised learning technique, k-means clustering. The inputs for training such networks are the time of the day, month of the year, location, and the type of criminal report. The goal is to allow LEAs to predict where, and at what time, a specific criminal activity has high probability of occurring.

4 Experimental Evaluation

Experimental tests were realized to measure and quantify the solution according to specific metrics. For each experiment 5 tests were executed for statistical analysis. The following metrics were chosen:

- **CPUconsumption** (Percentage): Percentage of CPU used by the overall tracking server solution.
- **Memory consumption** (Kilobytes): Numeric value of computer memory used by the overall tracking server.
- **Message lost rate** (Percentage): Percentage of messages not acknowledged (lost) by tracking server.
- **Maximum status delay measured** (Milliseconds): Maximum measured time a status message takes, from the mobile app status fetching process, to the successful introduction of data into the tracking server database. It is measured using the status message timestamp against the current database timestamp.
- **Network Traffic** (Kilobytes): Total sum of Bytes in and out of tracking server network interface.

4.1 Testbed

Mobile Network coverage and GPS signal strength are two uncontrollable variables with critical impact on testing. In order to eliminate these dependencies, the tests are run in a controllable local area network (LAN), and the GPS coordinates are generated internally by the App code. The mobile device app core (without user interface) was migrated to a single application that can manage multiple instances of mobile apps. The tracking server machine CPU has 2 cores at 2.0 GHz and 4 GB of RAM. It is connected to tracking units machine by a 100Mbps Ethernet link, with a measured latency lower than 1 ms. Enabling the clock synch (through Network Time Protocol, NTP) between machines with a much higher accuracy [6], measured offset is lower than 50 ms.

Every test starts by rebooting the tracking server machine, following a clock synch. A script that measures all metrics, launched on background, writes periodically (5 s) the corresponding values. The same script is also responsible to create the database table to store values, along with the generation of 100 users. Once the tracking server solution is launched, a new client is created and consequently connected to tracking server every 30 s. The evaluation is performed over the following scenarios: (1) TCP, no security; (2) UDP, no security; (3) UDP, with encryption (DES); (4) UDP, Encryption (DES) and MAC.

4.2 Experimental Results

The results are organized by metric. As shown in Fig. 4-a, not only the average CPU usage increases with the number of clients, but also the value of the CPU spikes. As expected, the addition of security functions originates additional CPU consumption.

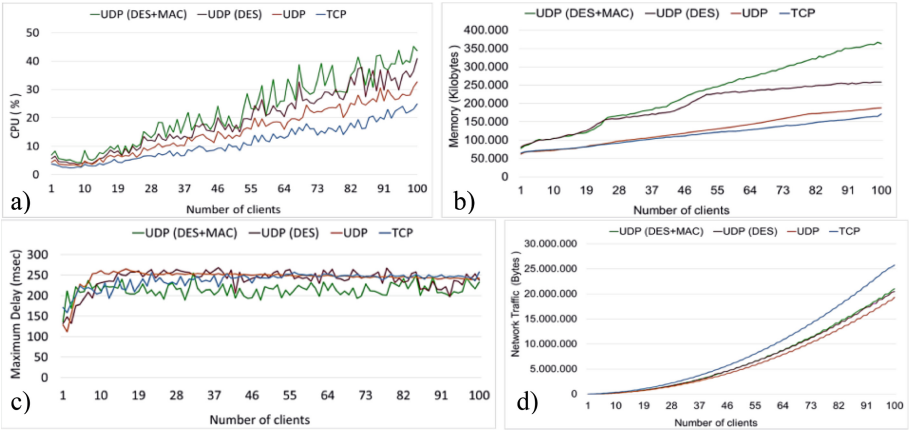

Fig. 4. (a) CPU consumption; (b) Memory consumption; (c) Maximum status delay; (d) Network traffic.

CPU average use increases 4.7 % whenever encryption (DES) is added to UDP, and 6.9 % with the addition of encryption along with MAC. Memory usage (Fig. 4-b) increases with the number of clients. The addition of security functions corresponds to additional memory consumption. Average memory in use increases 54 % when encryption (DES) is added to UDP, and 75 % with the addition of encryption along with MAC. Nevertheless, these values do not represent real memory in use, but memory allocated to the server application. The release of memory allocated but not in use, is managed by JVM (garbage collector) and machine OS.

Maximum status delay, as shown in Fig. 4-c, has similar values for all the tests (assuming a minimum of 50 ms offset). The values converge into 250 ms, which is the tracking server storage frequency of the received live status messages.

As shown in Fig. 4-d, TCP is the top traffic generator. This is mostly justified by the additional header size comparing to the UDP, since the combination of the test bed network and tracking server overload is relatively low turning the probability of a network segment to be retransmitted very low. Another fact that pushes TCP to generate more traffic is the overhead introduced by a connection oriented protocol. All together, makes TCP generate more 36 % traffic than UDP. Single encryption, and encryption with MAC, have similar values.

5 Conclusions and Future Work

The adoption of a RTMS specifically designed for LEAs has impact on daily performance of the organizations. However, it represents a large investment in tailor-made solutions, and eventual dedicated communication infrastructures (e.g. TETRA) for that purposes. This is an unfeasible alternative for a majority of countries with tight budgets. This work proposes a cost-effective RTMS that meets LEAs daily requirements with an extremely low budget, constituting an affordable solution that benefits from the

higher trend for the improvement of the existing mobile network coverage and corresponding terminal devices (e.g. Tablets). Our solution shows that it is feasible to trade computational resources for enhanced security on third-party networks such as mobile operators, in contrast to the creation of dedicated secure communication infra-structures such as TETRA, which come out to be too expensive and inadequate to meet the LEAs requirements pace (e.g. data throughput).

The proposed solution allows for the integration of more disruptive LEAs services and features on top of it. The embracement of specific LAE accessories, such as automatic license plate recognition, unmanned aerial vehicles, or even wearables like augmented reality glasses and smart wristbands, is also very promising for the evolution of the solution.

Machine learning services have shown to provide many benefits for adding intelligence to the Internet of Things [7]. These services, on top of the risk map feature, will enable vehicles to perform an optimized patrol, with better coverage of critical zones at the right time and weekday.

References

1. Qayyum, E., Mohsin, Z., Malik, J.: Real-Time Vehicle Tracking System Using GPS & GSM. Lap Lambert Academic Publishing, Saarbrücken (2013)
2. Varandas, L., Vaidya, B., Rodrigues, J.: mTracker: a mobile tracking application for pervasive environment. In: IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (2010)
3. Muruganandham, P.: Real Time Web based Vehicle Tracking using GPS. World Academy of Science Engineering and Technology, Los Angeles (2010). 37
4. Hasan, K., Rahman, M., Haque, A., Rahman, A., Rahman, T., Rasheed, M.: Cost effective GPS-GPRS based object tracking system. Lecture Notes in Engineering and Computer Science, vol. 2174, no. (1) (2009)
5. Chen, Y., Jia, W.: Challenge and solutions of NAT traversal for ubiquitous and pervasive applications on the Internet. J. Syst. Softw. **82**, 1620–1626 (2009)
6. Tomaciello, L., Vito, L., Rapuano, S.: One-way delay measurement: state of art. In: Proceedings of the IEEE Instrumentation and Measurement Technology Conference (2006)
7. Arsenio, A.: On the application of artificial intelligence techniques to create network intelligence. In: Laalaoui, Y., Bouguila, N. (eds.) Artificial Intelligence Applications in Information and Communication Technologies, vol. 607, pp. 71–97. Springer, Heidelberg (2015)