Fleet management and driver supervision using GPS and inertial measurements over GPRS networks

Cristian Paul Bara, Iulian Creţu, Ionuţ Roşu
Embedded Software Department
Arobs Transilvania Software
Cluj-Napoca, Romania
{paul.bara, iulian.cretu, ionut.rosu}@arobs.com

***Abstract*—The paper proposes a hardware solution to be equipped on automobiles for the purpose of fleet management and driver supervision. A very useful add on to vehicles. For fleets meant for very high reliability logistics, Just In Time or Sequence strategies. Tracking for insurance reasons to prevent fraud on both parts. The system uses GPS data and inertial measurements to track vehicle movement and driver discipline and sends data over a GPRS network to be logged. The software developed for this system communicates with the GPS transceiver and GPRS modem asynchronously. Application system management implemented using a call-callback method. The system is capable of Firmware Over The Air (FOTA).**

I. Introduction

The project started as part of the framework of CoMoSeF. The intended scope of the project is to domestically design and produce an advanced GPS tracking system for automobiles. Be it fleets or personal vehicles.

The architecture has been conceived on three levels: a hardware abstraction layer, a software development kit and an application layer. The application in developed specifically for a client's purpose with a high independence from the platform. Care has been taken in the process of programming the system to use non-blocking procedures, and more efficient parsing mechanisms. A large number of tasks are done in the background.

An adaptive tracking system has been implemented. GPS coordinates are sent on a time interval, a maximum distance, and on large course changes to smooth the trajectory in curves. An advantage of this system is that it is user configurable through SMS. The client can set motion and stationary tracking times, Firmware server IP address and port, Tracking server IP address and port, network configuration and list of accepted phone numbers.

Also implemented is a firmware update system over a GPRS network, or Firmware Over The Air (FOTA). The client does not need service the system, the tracker updates its firmware automatically.

II. System applications

A. General problems

It is a trend in the auto industry to keep up with smart phone development. Cars nowadays are expected to have a high degree of intelligence. It is unfeasible for automakers, or even buyers, to keep renewing technology at the same pace as our potable smart technology. Even more important is the fact that there are vehicles that do not tend necessarily to its driver. IT may be part of a fleet of vehicles meant for a specific task. It would be unfeasible to have sophisticated multimedia systems on such a vehicle. There is a need for the logistics planners to know what the vehicle is doing. Such a task is the purpose of our Tracking *plus* system[1].

Logistics planning is a very important part of any production cycle. With modern techniques of supply, like Just In Time or Just in Sequence strategies, it is very important to have a measure of control at a tactical level for logistic fleets. These methods were developed to eliminate the need for large warehouses. Manufacturers keeping supplies to last as little as 6 hours. More so for sorting facilities, for in sequence deliveries, it is vital that transportation networks are at peak performance. For this it is necessary to have a way of detecting faults in the delivery system as soon as possible or even ahead of time and reroute the vehicles [2][3].

Yet another problem is the wear and tear of the vehicle in the fleet. A great help would given if fleet managers can detect and take action on drivers with aggressive styles. A system like ours is capable of logging and report such problems. It is also noteworthy to mention that our system adapts itself by monitoring vehicle performance to determine a baseline. The user only needs to set a percentage level increase to detect problems.

Great importance should be given to gathering reliable data on vehicles in a fleet. It gives the possibility to planners to see unmodeled dynamics in the flow of events. This gives great insight in order to adapt the fleet's strategy in the future. Our system's data logging and transmission capability gives such an advantage. One can clearly see bottlenecks and other impedances in the process of vehicle transit directly on the map, in real time!

A new trend and in some places a necessity is vehicle tracking for insurance reasons. Advantageous for both parties because it leaves no doubt in determining the conditions of the events leading to accidents. The insurance firm may verify if the vehicle was used with common sense or prevent insurance fraud by the driver. The insurance holder may do the same to ensure that his point of view is valid and no inappropriate action is taken against him.

B. Specific problems and solutions

1) Real-time Dispatching and Dynamic Scheduling of Multidrop Route Planning and Dynamic Logistics Routing and Scheduling: On a strategic level one my determine that one

978-1-4799-1494-4/13/$31.00 ©2013 IEEE

transport per vehicle is inefficient. Our system, if set to log cargo loaded, carried and unloaded, can report to headquarters the state of transports. With this information one can reroute vehicled in the vicinity of a client. Thus quickening to a high degree response times to clients using real time data. The transport intended to pick up the cargo may suffer a defect, thus delaying transport, but no more!

2) Logistics Capacity Planning: Another future utility of such a system is sharing data between logistics companies to help intervene at a tactical level. Using such information the transportation company may judge in real time road congestion. The company may even subcontract other companies, with less demanding schedules, to take over certain transports of great importance. Thus the company may provide faster delivery times with a hypothetically unlimited supply of vehicles at its disposal. This way offering lower delivery times even considering unforeseen incapacitated vehicles.

3) Road works Vehicles Monitoring: Adapting the application of the tracker to measure parameters of a vehicle the user can track in real time, for example, where on a new road asphalt was poured. Meany other applications may be imagined, for instance a snow plow may be tracked to determine if it is clearing snow and if it needs assistance from other snow plows. Using information from other vehicles equipped with trackers the snow plows may be diverted to areas which are in danger of becoming congested. An up to date road works *news* service can be implemented.

4) Emergency Service Vehicle Dispatching: With a tracker installed on ambulances, police cars or fire trucks one does not need to enquire of the nearest vehicle to an accident, he has real time data of all the vehicles in the fleet and can dispatch the nearest free vehicle to the problem.

5) Car Theft: Drivers do not need to panic in case of car theft with our tracking system installed. The system reports periodically its position and may be configured by SMS to transmit more often or to call the police and report its position. The application can be configured to do this automatically if it is configured to know it should be stationary and it is on the move.

III. Present Developments

A. System Architecture

The system runs around a Software Development Kit (SDK) on which an application is written. The role of the SDK is to ease the development of the application, the higher level programmer does not need to know the specifics of the hardware used. There are Application Programming Interfaces (API) implemented for each subsystem, i.e. external flash memory, GPS module, GSM modem, etc. The APIs fall into two categories: Background tasks are accessed through the API only by retrieving results or by setting callbacks. The second type of API treats external hardware and micro-controller subsystems at the call of a specific function (external flash, etc.). There are also subsystems that run completely in the background (communication modules of the micro-controller).

B. Boot-loader

The boot-loader is a separate piece of code running in its own address space. It is the first in line on start-up and check for new firmware to be flashed onto the micro-controller. The boot procedure consists of checking for firmware in the external flash, checking for firmware integrity, flashing the new hardware and rechecking firmware integrity. If the procedure is successful the new firmware is run, otherwise a fail-safe firmware is flashed from the external flash, it is meant to run and download the firmware again.

C. Digital and Analog IOs

The GPS Tracker offers a series of digital outputs and digital and analog inputs to be used on the clients request. The intended uses of these pins are for checking for running state of the engine, presence of external power, measuring fuel gauges etc. A more specific example would be for the tracker to be mounted on road building equipment to measure when asphalt is poured and to add the location on the map.

D. GPS and Modem APIs

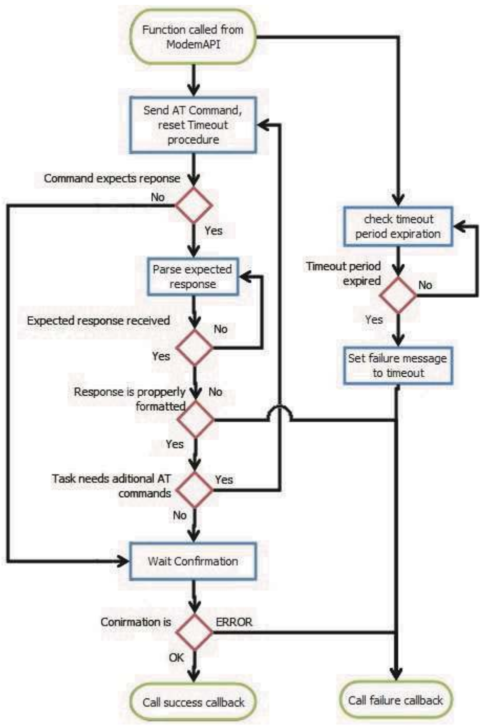

Fig. 1. AT command subsystem flowchart

Both GPS and Modem APIs have the role of retrieving data stored by their designated parsers. All parsers are implemented considering a well structured sentence from the GPRS and GPS modules. There are three types of messages NMEA messages from the GPS module and from the GPRS module AT and URC messages. All parsers work on similar principles. They are implemented as state machines implemented using pointers to functions [4]. Messages are never stored as a whole. The message is parsed character by character from a circular buffer. Once a message type is identified The specific message parser is activated. The parsers are non blocking. One character is parsed per application cycle. AT commands are received by request, URC messages are event based and NMEA messages are periodic [5][6][7].

In (1) we can see the AT command parser flowchart. The NMEA and URC parser are similar but simpler.

E. Basic FIR/IIR support

The SDK also offers a basic linear regression mechanism for implementing FIR/IIR filters so that these algorithms are not implemented in the application and in this way not creating delays. The application needs only to define a configuration structure for the desired filter and call the filter handler sending the pointer to the structure and the input data. The handler returns the result of the regression.

F. Interfacing over CAN bus

There is a CAN bus system implemented on the tracking device. Its purpose is to monitor engine data on vehicles equipped with CAN bus communication.

G. Application

1) State Machine: It is the role of the application to govern the operation of the Tracker through function calls to the SDK and reacting to callbacks from the SDK. There are two tasks defined in the SDK for the application. One is run on every pass through the main loop and the other is run at a fixed interval of time. At present the application treats external flash operations, Track server message generation, and Accelerometer [8], GPS and Modem data interpretation. It handles the adaptive coordinate transmission, it sends data on a time interval, distance interval, and on course changes greater than e specific amount. Also on specified time intervals the application check the Firmware Update Server for new firmware. All operations are done by state changes, the application does not execute complex operation on its own. During prolonged periods of vehicle inactivity The application enters a low power mode, powering on external modules only when necessary to preserve battery life.

2) SMS Parser: The role of SMS configuration is to provide the tracker with minimal configuration that will allow it to automatically update its configuration from an update server. Also there is the possibility to configure the device to use its basic functionalities as a tracker.

Minimal configuration options include:

- GPRS connection options like APN, user-name and password.
- Tracking server
- Configuration update server
- Tracking parameters like: tracking interval while stationary or when in motion

Basic device controls include:

- Reset device
- Reset configuration to factory defaults.

The device responds to the user with SMS messages reporting success or failure. While the device is in configuration mode all functionalities are disabled.

3) Firmware Over The Air:

a) Device software: This section will describe the software involved in FOTA that is located on the device.

There are two software components involved in this process:

- Boot loader
- Application firmware down-loader

b) Firmware down-loader: The firmware down-loader is a component of the Application. Its role is to check for new firmware on a firmware update server and download it to external flash. When the download is complete the component will restart the device, allowing the boot loader to write the CPU internal flash with the firmware stored in external flash.

c) Application update process:

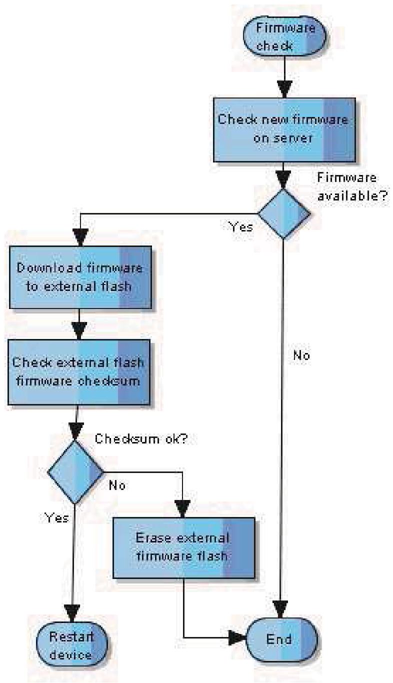

Fig. 2. Application firmware update flow chart

The diagram in (fig.2) describes the steps that the application will do for updating the firmware at a configured time interval. If the download was not successful then the external flash is erased and the processed will be retried after the configured time interval.

d) Boot loader update process: Its role is to check if there is a new firmware available in external flash and write it to CPU internal flash. If there is no firmware available it will launch the application stored in CPU internal flash. The following diagram describes the steps that the boot loader will do during start-up:

H. Simulators

There is also a SDK provided for development of the application level which allows the developer to test its functionality with or without a tracking system present. There are three levels of simulation: Software Simulation, Development board Simulation and Hardware Simulation.

The developer may chose to use a purely software to mimic the actual hardware. Another way to simulate is to use development boards for the peripherals. In this case the system architecture, except the hardware abstraction layer, resides on the developer's computer.

Hardware simulation uses the actual hardware to run the SDK, but the application layer is just a relay application. This is used to develop the application itself.

Simulations should in no way replace actual testing of the application on the tracker itself. Simulations are there to lessen actual programming of the micro controller's flash memory, preventing possible faults by overuse.

IV. Results

Our prototypes have been subjected to numerous real life tests. Course smoothing was tested more often to ensure reliability. In (fig.3) we observe course smoothing while passing through two 90 degree turns at slow speed.

Fig. 3. Course through intersection

Course smoothing has also shown to be working at higher speeds as well. In (fig.4) we can see that course changes can be detected even at higher speed, $50Km/h$.

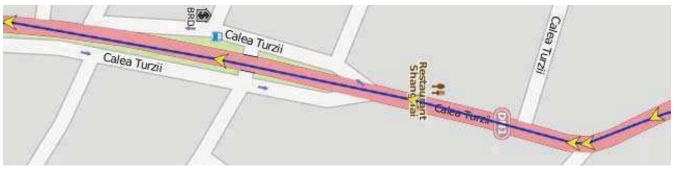

Fig. 4. Curve on *smooth* course

One of our prototypes has been subjected to a prolonged endurance test. It has been installed on a vehicle for approximately two month. During this time it did not have any considerable faults. It has also shown that SMS configuration is reliable. Actual configurations of data sent interval, course change sensitivity where done and also the timeout mechanism. Tests have had good results. The Firmware over the Air update system has also been tested in controlled conditions and has shown promising results. Saying this out *task management* system has proven to be performing to specifications.

V. Conclusion

Our product has shown endurance and reliability. It has also proven that can be integrated in a larger web of systems meant for automotive enhancements. It has numerous communication capabilities, wired or at a distance. A high processing power and connections lends to possibilities of vehicle *awareness*.

Automotive fleet managers need not renew vehicles for modern possibilities. Our tracker is a *drop-in*n piece of equipment offering modern facilities. Another step towards the *Internet of Things*.

VI. Future Developments

A. Configuration applications

The system will be provided with a web service client for configuration of the client's trackers. This application allows the user to manage the automatic updates of his devices. Trackers can be grouped into categories representing different applications needing different firmware versions. Once a device or category is set to update to a new firmware they will do so at the next set update event. The web service is in final development.

B. State estimation, Model identification

Further developments will include a greater performance in data conditioning and greater *trust* in the data gathered. State estimation will be used for *sensor fusion*. Data gathered from sensors and over communication systems will be conditioned through H_∞ state estimation [9]. This is needed for model uncertainty in, for instance, fuel consumption, or dead reckoning in care of GPS failure. System identification will be added as a background task. It will periodically renew data for the vehicle as a dynamic system. It will also relieve the customer from obtaining precise data regarding vehicle dynamics (e.g. fuel consumption based on engine performance) [10][11].

VII. Acknowledgments

We wish to thank the people also involved in this project without whos help the project may have not come to fruition. We thank conf. dr. Silviu Folea for designing the hardware the system is based on. We also wish to thank Mr. Vlad Timofte, our company's C/C++ system architect, for lending his knowledge to the project.

References

[1] D. Shapiro. Viewpoint: How mobile technology is accelerating the auto industry. [Online]. Available: http://analysis.telematicsupdate.com/infotainment/viewpoint-how-mobile-technology-accelerating-auto-industry

[2] H. Hirano and F. Makota, *JIT Is Flow: Practice and Principles of Lean Manufacturing*. PCS Press, Inc., 2006.

[3] S. M. Wagner and V. Silveira-Camargos, "Decision model for the application of just-in-sequence," 2009.

[4] M. Grosberg. Implementing efficient state machines. [Online]. Available: http://www.test.org/doe/

[5] u blox, "u-blox 6, receiver description, including protocol specification."

[6] U-blox, "Lea-6 - neo-6 - max-6 u-blox 6 glonass, gps & qzss modules. hardware integration manual."

[7] u Blox, "u-blox wireless modules, data and voice modules, at commands manual."

[8] Analog-Devices, "Adxl345 digirtal accelerometer documentation."

[9] I. Yaesh and U. Shaked, "Neuro-adaptive h_∞ estimation and its application to improved tracking in gps receivers," *Industrial Electronics, IEEE Transactions on*, 2009.

[10] A. Khodayari, A. Ghaffari, M. Nouri, S. Salehinia, and F. Alimardani, "Model predictive control system design for car-following behavior in real traffic flow," *Vehicular Electronics and Safety (ICVES), 2012 IEEE International Conference on*, 24-27 July 2012.

[11] W. Liu and G. Wang, "Auto-tuning procedure for model-based predictive controller," *Systems, Man, and Cybernetics, 2000 IEEE International Conference on (Volume:5)*, 2000.