SWARMON—Real-Time Localization System

Benoit Bourdon, Jean-Jacques Boye, Quentin Descours, Bastien Drouot, Olivier Reynet and Thibault Viravau

Abstract SWARMON is a project that started three years ago at ENSTA Bretagne. The aim is to provide a reliable positioning system to monitor and referee the WRSC. The WRSC is a competition involving small autonomous sailboats (50 cm–5 m), so any additional hardware has to be both small and light. The boats cruise near shore, hence the system can rely on cellular network for data exchanges. The system described in this article is based on both an Android application and a custom hardware tracker on the client side and uses a HTTP RESTful API on the server side. This server allows real-time tracking and replay mode.

1 Introduction

In 2013, a first version of the SWARMON system was developed and deployed during the WRSC2013 in Brest. It aimed at localizing the boats in real time during the WRSC events and to inform the competitors and the public by broadcasting a map on Internet.

This first attempt was based on an Android application running on a waterproofed smartphone and placed on the boat, a simple HTTP server and a dedicated JavaScript-based client to show the map on a screen. Even if this first attempt succeeded in localizing boats, many problems arose due to a lack of system engineering: the GPS coordinates of the boats were sent using SMS, which generated delays, the Android application worked differently depending on the smartphones, the client side application could only be used on one computer at a time and the database could not be accessed via the Internet.

Learning from this first experience, we enhanced the development based on the following requirements:

- The GPS tracking device should be as small as possible to fit in any sailing robot
- The localization process should avoid relying on SMS and use 3G or 4G instead

B. Bourdon · J.-J. Boye · Q. Descours (✉) · B. Drouot · O. Reynet (✉) · T. Viravau
ENSTA Bretagne, 2 Rue François Verny, Brest, France
e-mail: olivier.reynet@ensta-bretagne.fr
e-mail: quentin.descours@ensta-bretagne.org

A. Friebe and F. Haug (eds.), *Robotic Sailing 2015*,
DOI 10.1007/978-3-319-23335-2_9

- The data should be exchanged via Internet through a HTTP front-end
- The client side should only rely on HTTP requests (through a browser, an application or a script)
- The sailing robot should receive a notification when a collision may happen

Therefore it was decided to start a new project in order to:

1. Build a custom hardware tracker to localize the boats
2. Use a web framework to develop the services
3. Create a new Android application to keep using smartphones which are cheap and ubiquitous

The first part of this article explains the technical choices made. The second part describes the SWARMON global system, which consists of a GPS tracker and an Android application interacting with an HTTP server. The third part gives results of the system testing. The last part discusses the possibility of implementing a collision avoidance system.

2 Technical Choices

The key point to keep in mind is that this entire project is the result of successive student projects conducted at the ENSTA Bretagne. Thus, the technical choices were mostly made in regard of their pedagogical potential.

Concerning the development of a device to localize the sailing robots, many solutions have already been developed in the past few years. The development of a custom AIS[1] device (the commercial receivers being too expensive and bulky) has been a considered possibility. However, the students involved were more interested in web development and software engineering than telecommunication. That is why we preferred a web-based approach and chose to base our work on a 3G + GPS module instead.

This web-based approach has also been decided because of the numerous technologies involved. For instance, the use of a web framework allows the iterative creation of a complex system by getting in touch with several concepts and languages.

3 SWARMON Global System

The SWARMON global system is described in the Fig. 1.

The tracking device can be one of the following: a smartphone using our Android application, our custom tracking device based on a 3G + GPS module or any system compliant with our RESTful API. The GPS coordinates (latitude, longitude and GPS timestamp) are gathered and stored on the server to allow later uses. The user can monitor the robots in real time or use the replay mode to display coordinates from

[1] Automatic Identification System.

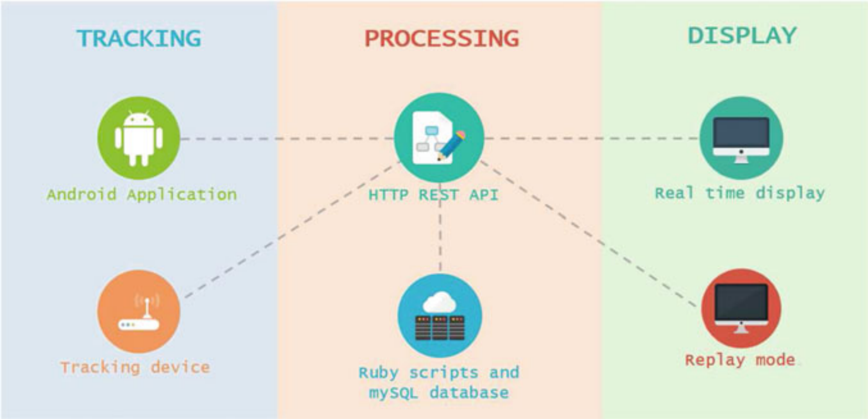

Fig. 1 Description of the SWARMON system

previous events. In both cases, the name of the robot, its team and its category can be displayed directly on the screen with a dedicated colour scheme. The organizers of the competition can create beforehand informative markers which will then be displayed in real time, allowing the public and the teams to identify buoys, obstacles or any area of importance for the current event.

3.1 The Ruby on Rails Web Framework

The server uses Ruby on Rails, an open-source web application framework based on the Model-View-Controller design pattern [1]. This framework has been chosen because it is both an effective and pedagogical tool. By using it, the students have discovered many web standards and technologies but have also gotten in touch with software engineering patterns such as the concepts of 'Convention over Configuration', 'Don't Repeat Yourself' or the 'Active Record Pattern'.

This code and all the hardware description are available on GitHub.[2]

3.2 An Universal and Secured Front End

To interface a GPS tracking system, we chose to develop a HTTP RESTful API [2]. Efficient web services are built using HTTP methods GET, PUT, DELETE and POST [3] and the concept of resources. For instance, in order to push a GPS coordinate into the database, the interaction between the tracker and the server takes

[2]https://github.com/olivierreynet/WRSCMonitor/tree/master.

URL	GET	POST	PUT	DELETE
/real-time	Visitor			
/replay	Visitor			
/session	Logged in	Logged in	Logged in	Logged in
/myaccount	Logged in	Logged in	Logged in	Logged in
/members	Visitor	Visitor	Logged in	Logged in
/teams	Visitor	Logged in	Team Leader	Team Leader
/robots	Visitor	Team Leader	Team Leader	Team Leader
/missions	Visitor	Team Leader	Team Leader	Team Leader
/attempts	Teamate	Team Leader	Team Leader	Team Leader
/trackers	Admin	Admin	Admin	Admin
/coordinates	Teamate	Tracker	Admin	
/what	Android			
/getCurrentMission	Android			
/getMissionInfos	Android			
/gatherCoordsBetweenDates	Android			
/gatherCoordsSince	Android			

Fig. 2 HTTP REST API of the server side, roles and associated permissions

the form of a HTTP POST request on the URL http://www.myserver/coordinates. Another example : if a client wants to get the list of all the sailing robots engaged in the competition, it just has to send a HTTP request using the GET method on the URL http://www.myserver/robots.

The complete HTTP RESTful API is given in Fig. 2. For each HTML request on each URL, the database behaviour can be scripted and the response format can be either HTML or JSON. When using a browser, a simple graphical user interface allows the users to Create, Read, Update and Delete (CRUD) any data, provided that they possess the required permission. As it is described in the Fig. 2, each user of the system has a specific role to which permissions are granted.

To secure the server side, the system implements the HTTP Basic Authentication method to authenticate the users of the system whereas the HTTP Token method is used to authenticate a tracker (hardware or Android Application). Users (referees, sailing teams) have to register and log in to perform some operations (creating a robot, linking a tracker to a robot) while race live display and replay are public. The trackers are authenticated with a token generated by the server which has to be sent by the tracker in each HTTP request.

3.3 Hardware Tracker

The GPS tracking device has been built around the SIMCOM SIM5218E module which provides both GPS and 3G capabilities (Fig. 3).

This module runs a Lua script to retrieve GPS coordinates. It also sends these coordinates to the server and stores them on an external SD card. The module stores the data even if there is no 3G coverage in the current area or if the server is unavailable. The development hardware provided by the maker was useful for the tests, but in order to reduce the size of the tracker, a dedicated interface board has been built, which only embeds the needed features for our use of the module (USB/UART, battery, SIM and SD). Alongside this board and the module, the tracker also includes a GPS antenna, a GSM antenna and a battery (2000 mAh, 3.7 V). The tracker can retrieve up to 10 positions per second, this rate is configured in the Lua script operating the tracker. We choose to use a 1 position per second refresh rate because a higher rate is of little use considering the latence of the 3G network and the relatively low speed of the boats. With this configuration, the tracker achieves an autonomy of 8 h.

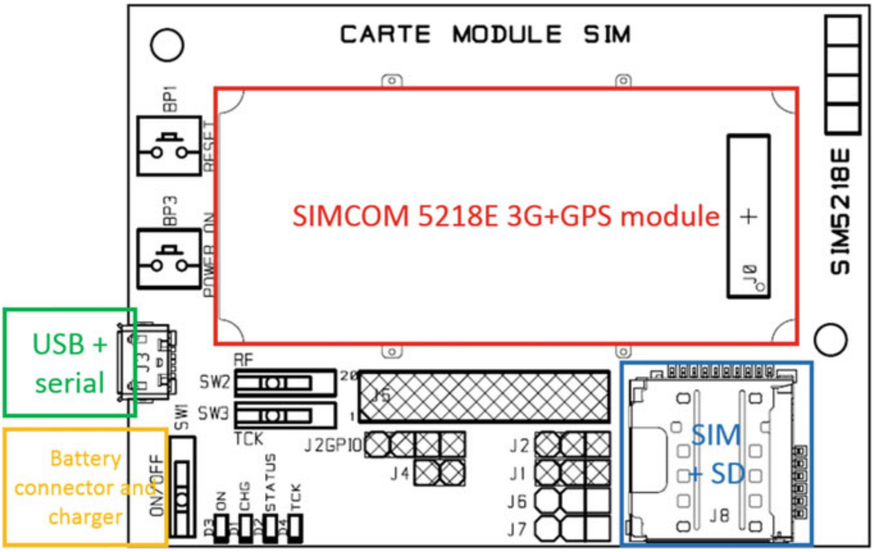

Fig. 3 Electronic description of the SWARMON tracker based on SIMCOM 5218E

3.4 Android Application

A minimalist Android application has been developed to offer an alternative tracking system. This standalone application makes use of the technologies embedded in all modern smartphone to track the sailing robots during the competition : 3G network is used to send the coordinates to the server by using the same API as the tracker. The data are stored in the phone's memory thanks to a local database. In case of a 3G connection loss, the data can be sent to the server afterwards. In Fig. 4, this Android Application is described in terms of activities and processes. Even if the Home button is pressed, the service keeps tracking position and sending messages thanks to a background service.

Fig. 4 Description of the Android application in terms of activities and processes

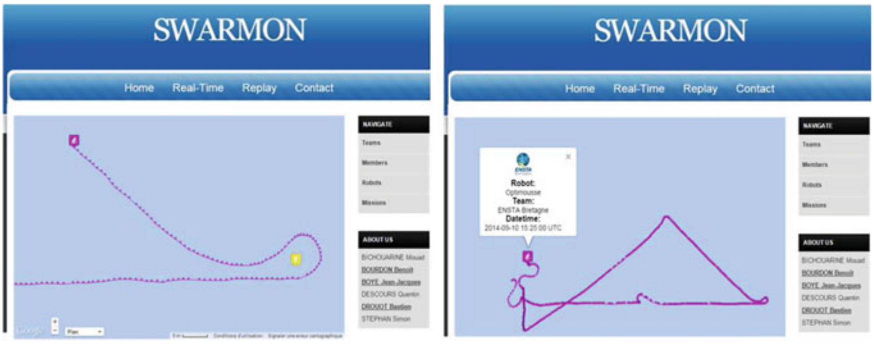

Fig. 5 Real time display (on the *left*) and Replay mode (on the *right*) during the WRSC 2014 in Galway

4 Sea Trials and Testing

In September 2014, an early prototype of the device has been tested on the robots of the ENSTA Bretagne during the WRSC 2014. Even far from the coast of the Galway harbour, the system performed perfectly well and did not suffer from any signal losses. The two modes of display offered by the website at this time are shown in Fig. 5.

In March 2015, tests have been conducted in the Brest harbour (see Fig. 6) to confirm the effectiveness of the GPS tracking device, the Android application and the web application. During these tests, four different Android smartphones and two trackers have been used at the same time to retrieve GPS coordinates, and four devices were accessing the live display application. The server performed as expected by receiving and storing the coordinates in the database. The users were able to follow the testing in real time and in replay mode.

The GPS tracking devices relying on custom hardware also completed the tests and performed better than the smartphones (more accurate, higher refresh rate, more consistent transmission rate). This is due to their two external antennas and their higher reception rate of one position per second (smartphones can retrieve positions every 3 s at best).

To test the Android application we used four models from three different makers.[3] The result is that the application relies heavily on the quality of the smartphone's hardware but also from software optimizations from the maker to improve the autonomy. Even if some models are rather close to the tracking device in terms of accuracy and stability,[4] some others suffer from frequent signal loss or really poor accuracy of the embedded GPS receiver.[5]

[3] Sony Xperia Z1, Wiko Cink Peak, Wiko Ozzy, Nokia X.

[4] Wiko Ozzy.

[5] Wiko Cink Peak.

Fig. 6 Geolocalization of two SIMCOM tracking devices (in *black*) compared to a differential GPS (in *red*) during the sea testing

Despite the fact that the accuracy of the GPS position strongly depends on the smartphone, the Android application was stable and completed every implemented features.

5 Possibility of a Collision Avoidance System via HTTP Responses

Now that the tracking system is fully operational, a collision avoidance system can be implemented in SWARMON:

- All the data processing is made in real time on the server side. The latency of the tracking architecture does not exceed a few seconds, which is few compared to the time scale of the sailing robots shiftings.
- The tracking devices provide a serial and/or a USB link which can used to communicate with the sailing robot.
- Each time a tracker posts its localization to the server, the HTTP server responds to the tracker with a HTTP code or a body response[6] to pass information to the tracker.

[6]cf. https://en.wikipedia.org/wiki/List_of_HTTP_status_codes.

Advanced behaviors can then be implemented on the server side or on the sailing robots, thanks to the serial link between the 3G connected tracker and the robots that can be used to exchange data. Rather than just sending an alert, which could lead to another hazardous situation, depending on the drones reaction, the server could describe to each boat the safest route to sail in order to definitely avoid a collision. The server may also suggest the best route to take, depending on the wind, the tides or the ocean currents.

This centralized architecture offers considerable possibilities as already demonstrated by papers from previous WRSC proceedings [4, 5].

6 Conclusion

The SWARMON project is an efficient solution to monitor connected objects. New services as virtual boundaries or swarm behaviours can easily be integrated in this architecture, because all the required tools to compute the data or to communicate with the robots already exist. The only remaining limitation of the system is due to the use of an internet connection which currently restricts its use to coastal environments and prevents any use in open sea. In this configuration, an interesting solution would be to develop a dedicated device based on the AIS system, using the undefined AIS messages to transmit drone and swarm specific information (collaborative behaviours, data exchange, orders).

7 Credits

- N. Brocheton, SWARMON Swarm Monitoring for WRSC 2013, ENSTA Bretagne, 2013
- M. Bichouarine, B. Bourdon, J-J. Boye, Q. Descours, S. Stephan, SWARMON, ENSTA Bretagne, 2014
- B. Bourdon and B. Drouot, Monitor Your Robot: Dedicated monitoring web site for WRSC, ENSTA Bretagne, 2015
- T. Viravau, Android Tracking Application for WRSC, ENSTA Bretagne, 2015
- S. Hunault and T. Zheng, Collision Avoidance System and Autoscoring Capabilities for WRSC, ENSTA Bretagne, 2015

References

1. Ruby S, Thomas D, Hansson DH (2013) Agile web development with rails 4. Pragmatic Bookshelf
2. Richardson L, Amundsen M, Ruby S (2013) RESTful web APIs. O'Reilly Media

3. Gourley D, Totty B, Sayer M, Aggarwal A, Reddy S (2002) HTTP: the definitive guide. O'Reilly Media
4. Ammann N, Biemann R, Hartmann F, Hauft C, Heinecke I, Jauer P, Krger J, Meyer T, Bruder R, Schlaefer A (2010) Towards autonomous one-design sailboat racing: navigation, communication and collision avoidance. In: Proceedings of the 3rd international robotics sailing conference
5. Ammann N, Hartmann F, Jauer P, Krger J, Meyer T, Bruder R, Schlaefer A (2011) Global data storage for collision avoidance in robotic sailboat racing theworld server approach. In: Proceedings of the 4th international robotics sailing conference