Proceedings of MASPLAS'04
Mid-Atlantic Student Workshop on Programming Languages and Systems
Seton Hall University, April 3, 2004

Integration of GPS Devices with Computer Software for Airline Safety Applying C/C++ and Perl

Jon Ellch, Michael Carter, Sean Henderson, Michael Heenan, Ruijian Zhang
Purdue University Calumet

Abstract

The horrible events of September 11, 2001 unveiled a devastating realm of vulnerability and inefficiency in air travel tracking and safety. In this research, we pursue the integration of Global Positioning System (GPS) devices with computer software applying C/C++ and Perl. This paper offers a comprehensive solution that will minimize risk and maximize fault tolerance in the most vulnerable targets. Our project offers a universal technique for tracking planes on the ground and in the air. It provides a supplement to the existing radar system. It prevents mid air collisions and the use of planes as weapons. It prevents runway incursions/ground collisions. The system also ensures fault tolerance for the air traffic control tower. Additionally, the project deploys system wide "No Fly Zone", and "Soft Wall" alert and support.

The project can be categorized into two types of program: non-user interaction and user interaction. The former is responsible for transmitting data to the database. These were written in PERL because of its simple database interface and superior string manipulation ability. The use of PERL also made these programs cross platform. The latter needs both speed and GUI's. The need for speed encourages us to use C/C++ instead of Java or PERL. We use C to modify the gps-drive. The three dimensional graphical view reaps the benefits of object oriented design using C++ and OpenGL.

1. Introduction

The horrible events of September 11, 2001 unveiled a devastating realm of vulnerability and inefficiency in air travel tracking and safety. Since September 11th, the issue of safety remains haunting and intense—even the bravest patron and pilot question the ultimate fate of surviving a flight. This paper offers a comprehensive solution, which addresses the reality that the government and airline industry face a relentless challenge to demonstrate perpetual efforts and results ensuring air travel safety. In this research, we pursue the integration of Global Positioning System (GPS) devices with computer software applying C/C++ and PERLl. The system creates an efficient and diversified method for air travel tracking and safety. We have identified that risk factors are prevalent not only in the air, but also on the runway and within the control tower. The objective of our project is to offer a solution that will minimize risk and maximize fault tolerance in the most vulnerable targets. The chief factors to such a scheme involve:

1) Identifying air travel activity on no-fly zones;

2) Utilizing the system to prevent runway incursions/ground collisions; and

3) Ensuring fault tolerance for the air traffic control tower.

This paper is organized in five sections. Section Two describes the motivation and objectives of our project. Section Three elaborates on the system design and our strategic approach. Section Four elucidates the implementation of the system and the technology usage. Finally, Section Five provides concluding remarks and future directions.

2. Motivation and Objectives

Our goal is to ensure that the project is beneficial, innovative, and cost efficient. A major objective of our project is that it should offer a universal technique for tracking planes on the ground and in the air. The National Transportation Safety Board (NTSB) has made recommendations for its "Most Wanted Transportation Safety Improvements Aviation Issue Areas." One key area of concern is runway incursions/ground collisions of aircraft. Current systems either do not have support for runway collision avoidance, or they track aircraft on the ground and in the air separately. Another objective is that the project should provide a supplement to the existing radar system. In essence, it addresses the risk of planes going "off radar." The need is imperative because it prevents mid air collisions and the use of planes as weapons. Additionally, the system deploys system wide "no fly zone", and "soft wall" alert and support. Presently, this matter is being addressed as a stand-alone system for each entity.

Also, the system should provide the following benefits:

• a three-dimensional graphical view to enhance visibility;

• a continuous monitoring tool through a simulated glide path tunnel to assist pilot navigation;

• a more efficient and dependable manner of retrieving the plane's data through the use of a database; and

• a comparative analysis tool for all air travel activity.

GPS devices have been utilized in airplanes long before the issue of air travel safety intensified. Many commercial projects are designed to track aircraft or prevent collisions—TCAS (Traffic Alert and Collision Avoidance System) is one of them. But our project offers a unique approach to provide avionic tracking and safety. What makes our project unique is its integration and modularity. It offers a solution that incorporates tracking distinctions and efforts into one entity and creates an alternative to stand-alone approaches. Presently, the airline industry and government address flight tracking, in-air collision avoidance, and runway collision avoidance independently from each other. There is no evidence that a coherent system integrating the problems and solutions of the aforementioned exists at this point.

Another concept of uniqueness evolves around the usage of fault tolerance. Recent advances in database technology allow for redundant systems to automatically take over if the primary database should fail. Furthermore, if any component of our system fails, it has only a negligible effect on the system as a whole. For example, if the 3D-client failed to operate, other infrastructures would not be affected, and the default to 2D-client would allow a continuance of full operation without interruption. This is imperative for disaster prevention at the control tower interface.

In addition, our system makes efficient use of existing infrastructure. Current systems use expensive proprietary networking hardware. We are proposing the use of the current nationwide 3G cellular networks. Modern cellular networks are capable of moving data at a rate of 100 kbps, which is well within the limits of what our program needs. This alleviates the need to create expensive infrastructure, similar to the current network of radar stations, solely for the use of tracking planes.

3. System Design

The system design is shown in Figure 1. The following is a synopsis of each component's function:

• The heartbeat of the project is a database. None of the other programs can function without it. Every other application either puts data in or extracts data from the database.

• The 2D-client is the application the user will most likely to interact with. The 2D-client is a user-friendly program capable of displaying data extracted from the database and the positions of the planes on top of a two dimensional map. The 2D-client will also be responsible for alerting the users about critical flight events, such as a plane off course or flying below the radar, etc. The 2D-client stems from the popular open source program GpsDrive.

• The 3dclient will display the same data from the database as the 2D-client, however, it does not provide an easy method to prompt the user for input. Consider the 3D-client, an accessory program running on an additional monitor next to the 2D-client, providing a more in depth view of a plane's location using OpenGL.

• The modified GpsDrive and the 3D-client will receive events from the Logic Server and respond accordingly by alerting the user about a problematic situation.

• The Logic Server is a program that analyzes the database continuously and redlines problems. Once the Logic Server detects an event, it notifies all interactive clients.

• The Real Client represents a small program that translates the NMEA (National Marine Electronic Association) output of the GPS-hardware into the internal database format, which continuously updates to the database.

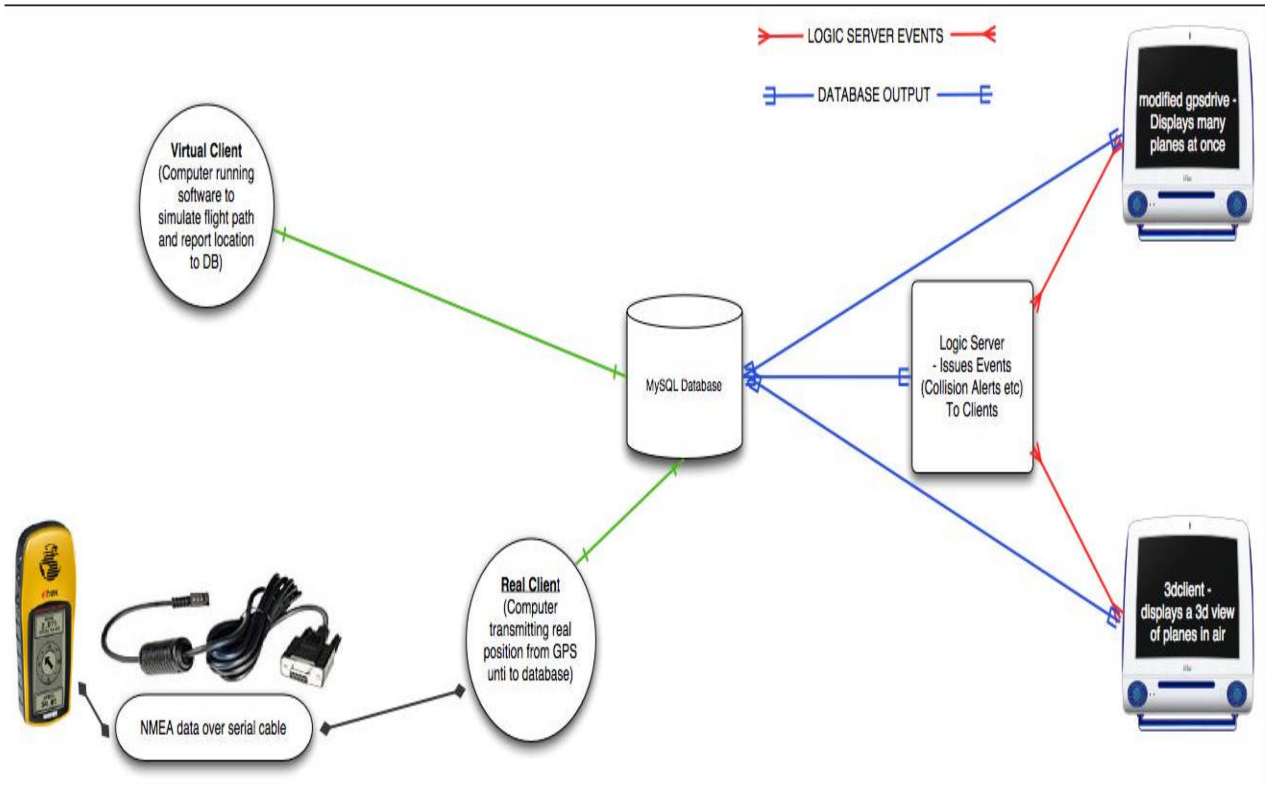
• The Virtual Client will be a slightly larger program that updates the database continuously with a virtual position—emulating the flight plan of a plane. Many instances of the virtual client can be driven by scripts to allow the emulation of complex situations involving more than one plane.

Let us see the flow of the GPS data first. The GPS data originates at the eTrex handheld GPS unit and travels over the serial cable into our laptop, which represents the computer in a real plane. At this point the "Real Client" is dubbed. At a specified interval, the real client will transmit the data over a wireless network and insert it into the database. Once in the database, the 2D-client, 3D-client, and Logic Server programs will use the data. We wrote a "Virtual Client" program that can simulate one or more of the planes in the database for debugging purposes.

We have the 2D- and 3D-clients constantly extracting data from the database. These are the interfaces designed for user interaction. The programs are responsible for displaying information about the planes' positions and reporting their status in an intuitive manner. They also alert appropriate parties when "events" happen, for example, the 2D- and 3D-clients both should notify the user.

The logic server is designed to notify any client (2D-client / 3D-client) whenever an event happens. Using an independent program that communicates over the network has many advantages. One of most important advantages is that it reduces duplicate code and work. The 2D-client and the 3D-client do not analyze the database constantly. Instead, the clients use the logic server to analyze and alert. Another key advantage is that the independent program allows a conversion of a definition to ensure that it can be understood system wide and individual clients have the same settings.

Figure 1. System Design


LOGIC SERVER EVENTS
DATABASE OUTPUT
Virtual Client (Computer running software to simulate flight path and report location to DB)
MySQL Database
Logic Server - Issues Events (Collision Alerts etc) To Clients
modified gpsdrive - Displays many planes at once
3dclient - displays a 3d view of planes in air
Real Client (Computer transmitting real position from GPS unti to database)
NMEA data over serial cable

The design makes development faster since the project can be partitioned into many self-contained modules that can be developed concurrently. It also allows for expandability. If we need to add more events or another client to display or analyze the data in a different way, the infrastructure is already present.

4. Implementation and Language Features

We applied C/C++ and Perl as programming languages to implement the system design. We developed an integration of Global Positioning System (GPS) devices with computer software

The project can be categorized into two types of program: non-user interaction and user interaction. The former is responsible for transmitting data to the database. These were written in PERL because of its simple database interface and superior string manipulation ability. The use of PERL also made these programs cross platform. The latter needs both speed and GUI's. The need for speed encourages us to use C/C++ instead of Java or PERL. We use C to modify the GPS-drive. The three dimensional graphical view reaps the benefits of object oriented design using C++ and OpenGL.

All the information about a given plane's position is stored in a database so that our system can display the plane's location and make sure that it is safe. If the plane is not safe, the system warns the user of the problem. The user will have both a

2D-client and a 3D-client to work with. The 2D-client is built on an open source program GPS-drive. It shows the user any warning messages that might occur along with a view of planes' positions on top of a two dimensional map. The 3D-client is an accessory to the 2D-client. It gives the user a more in depth view of a plane's location using OpenGL. The logic server analyzes data in the database to check for possible hazards such as two planes being too close and broadcasts any warning messages to all clients connected. The user only sees the results of the logic server via the 2D-client or the 3D-client; they never interact with the logic server. The system uses wireless networking to allow the clients to communicate with the logic server and the database. Data originates from eTrex handheld GPS units and is entered into the database through a serial cable connected to a laptop. This allows for multiple GPS units to be working simultaneously from various locations. Our system is an inexpensive and portable approach to making the world a safer place.

5. Conclusion

In this paper, we designed and implemented a computer-based solution to a real world problem. Our system uses software along with GPS devices to make airline travel safer. The main goal is to prevent unnecessary collisions of aircraft with other aircraft, or with buildings.

All the information about a given plane's position is stored in a database so that our system can display the plane's location and make sure that it is safe. If the plane is not safe, the system warns the user of the problem. The system uses wireless networking to allow the clients to communicate with the logic server and the database. Data originates from eTrex handheld GPS units and is entered into the database through a serial cable connected to a laptop. This allows for multiple GPS units to be working simultaneously from various locations. Our system is an inexpensive and portable approach to making the world a safer place.

Reference:

[1] National Transportation Safety Board (NTSB) www.ntsb.gov/aviation/aviation.htm

[2] Dale Depriest, "A GPS User Manual: Working With Garmin Receivers", The National Marine Electronics Association: 2003, page 225.

[3] GpsDrive: gpsdrive.kraftvoll.at/index.shtml

[4] Jeff Molofee, "A Windows Base Application for Unix Model", 2003.

[5] www.mysql.com