An Internet-Based Interactive Embedded Data-Acquisition System for Real-Time Applications

Ali Ziya Alkar, *Member, IEEE*, and Mehmet Atif Karaca

***Abstract*—In this paper, we present the principles of a low-operational-cost but flexible Internet-based data-acquisition system. The main core of the system is an embedded hardware running a scaled-down version of Linux: a popular choice of operating system for embedded applications. The embedded device communicates through General Packet Radio Service (GPRS), which makes it accessible from anywhere in the world through a web server built into the embedded device. In addition, GPRS provides a bidirectional real-time data transfer allowing interaction. The proposed system eliminates the need for server software and maintenance. A novel approach is introduced to minimize the operational costs while operating with a large amount of data. The system is demonstrated to be suitable for different embedded applications by attaching several real-time modules through appropriate interfaces.**

***Index Terms*—Data-acquisition, embedded system, interaction, Internet, Linux, real time.**

I. Introduction

DATA-ACQUISITION systems with remote accessibility are in great demand in industry and consumer applications. In some applications, human beings have been replaced by unmanned devices that will acquire data and relay the data back to the base [1]. There are data-acquisition and control devices that will be a substitute for a supervisor in a multisite job operation. A single person can monitor and even interact with the ongoing work from a single base station. An acquisition unit designed to collect data in their simplest form is detailed in [2], which is based on Linux [3], which is a popular choice for embedded PC systems. A similar system in [4] provides data-acquisition with no concern for remote access. Data collection for postprocessing on a vehicle's position for an advanced traffic survey is discussed in [5].

Some applications adding remote accessibility are detailed in [6] and [7], which are built to collect and send data through a modem to a server. Although these are well-built systems that serve the purpose for a specific task, the user cannot interact with the system. Another unidirectional data transfer is presented in [8], which uses the Global System for Mobile Communications (GSM): a popular wireless choice for connectivity between the data-acquisition units and clients.

There are also several systems that allow data to be remotely accessed. As a solution to wireless data collection through the Internet, General Packet Radio Service (GPRS) is a popular choice in several applications. A surveillance system based on GPRS is presented in [9]. A recent work [10] has presented a GPRS solution to the data-acquisition problem for remote areas. A distributed system capable of road vehicle locating, monitoring, and telemetering with GPRS is presented in [11]. A long-distance data-collection system for the Earth tide gravimeter, collecting information on temperature, humidity, atmospheric pressure, etc., is designed with GPRS using a hard-coded static Internet Protocol (IP) address [12]. These systems use GPRS without concerns about minimizing the cost of data transfer.

Similar types of Internet-based systems, such as those in [13]–[16], are designed to gather a bulk of data before serving them upon request. In these applications, data are compiled in a central server and are then served to the clients via the Internet. The client framework is in a central server and has all the applications. A person that needs to access any data must first access the server. An indirect access to the data-acquisition unit makes the system unattractive for real-time control applications, where direct interaction with the system may be required. The need to maintain an additional server will also increase the setup costs and the costs to maintain the acquisition systems, such as regular maintenance costs, system updates, etc.

Therefore, the central server has to be eliminated for a real-time system. The closest to this idea is published in [17]. In this system, a reliable bidirectional Point-to-Point Protocol (PPP) link for real-time control and surveillance via a GSM network is formed. However, there is still no effort to minimize the operational costs (including the costs to transfer a large amount of data). In addition, this system is based on an industrial PC, thus making it an expensive solution.

Interaction with the embedded unit is also an important issue. In [18], an embedded PC card placed on the Internet allows limited interaction through commands sent through Transmission Control Protocol/IP (TCP/IP) and User Datagram Protocol.

In this paper, we propose a GPRS-based portable low-cost data-acquisition system, which can establish a reliable bidirectional connection for data-acquisition. The proposed system uniquely reduces the costs occurring from frequently requested data and eliminates the need for a well-established server. The system uses a dummy server for static information, thus optimizing the transfer of large data. The user can directly log in and interact with the embedded device in real time without the need to maintain an additional server. The system is modularly built, allowing different modules to be added. In addition, it is

Manuscript received November 16, 2007; revised July 10, 2008. First published September 30, 2008; current version published February 9, 2009.

A. Z. Alkar is with Hacettepe University, 06800 Ankara, Turkey (e-mail: alkar@hacettepe.edu.tr).

M. A. Karaca is with Sistemim Electronics Design and Consulting Company, 06800 Ankara, Turkey (e-mail: mehmet@sistemim.com.tr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIM.2008.2005278

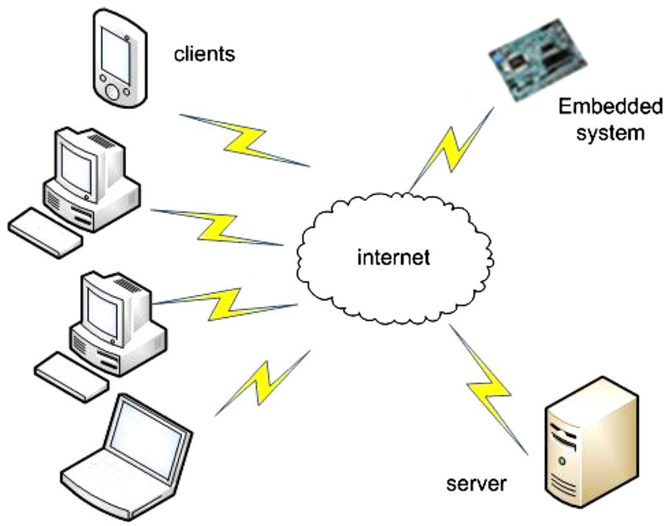

Fig. 1. General diagram of a data-acquisition and control system.

flexible to accommodate a wide range of measurement devices with appropriate interfaces.

In Section II, we will introduce the details of the aforementioned system. In Section III, we will provide an example that will introduce some of the capabilities of the system using a collection of modules. In this paper, a camera, a Global Positioning System (GPS), and a temperature chip are connected to the embedded system to demonstrate its operation. The operational features are presented, which are needed to be considered for practical applications. Section IV presents the conclusion.

II. Interactive Data-Acquisition System

The general principles of Internet-based control systems have been modeled in [19]. Interactive Internet-based systems provide a way to monitor and adjust using standard web browsers and a PC. The target systems can be monitored and controlled independent from the location and the platform since standard web browsers can be used on the client side.

A typical data-acquisition system is made up of three components connected to each other via the Internet, as shown in Fig. 1. The data-acquisition system needs to relay the acquired information to the requesting clients. The clients also need to send commands. If necessary, this is implemented through a server, and then, an enormous amount of data transfer time would be consumed. Thus, alternative methods need to be explored.

A. Establishing a Direct Communication Link Between the Client and the Embedded Device

GSM and GPRS [20] are developed for cellular mobile communication. A GPRS connection with unlimited duration of connectivity is charged only for the data package transfers and adopted in several mobile remote control/access systems [13], [14], [16]. GPRS becomes a cost-effective solution only if the data transfers can be optimized.

Once a GPRS connection has been established, queried data can be relayed to the client via a central server [13]–[16]. Using a central server to relay the acquired data has some disadvantages. First, a central server needs a client interface framework. An additional data transfer corresponds to time delays before the data are made available to the client. In addition, since the server acts as a relay, no direct bidirectional communication between the client and the embedded system can be established. This makes the system unsuitable for real-time control applications. The basic idea behind real-time processing is that the embedded system is expected to respond to the queries in time. Real time should be fast enough in the context in which the system is operating and reliable as well. Real-time system correctness depends not only on the correctness of the logical result of the computation but also on the result delivery time [21]. This method also increases the data transfer cost as the number of clients increases due to the access amount of data transfers via GPRS.

Direct communication, on the other hand, enables access to only relevant information in the embedded system by preprocessing the data. The embedded system should also handle the web services. This eliminates the need for a central server and reduces the amount of data sent from the remote unit since only the queried data will be transferred.

In the proposed system, the GPRS architecture and protocols are compliant with [20]. This system is configured to be virtually online at all times in a GSM network. An admin script is executed after the boot of the operating system, initiating the GPRS connection software module. A PPP connection is established by a GPRS modem that works at 900/1800/1900 MHz operating frequencies. A PPP daemon (PPPD) is used to manage the PPP network connections between the client and the embedded module. The PPPD is responsible for setting up the GPRS parameters, such as the connection speed and compression.

To directly access an embedded system, the IP address of the embedded device should be made available to the client side. There are two choices available. A static (hard-coded) IP could be used, or the remote device should initiate a connection by reporting its IP. This choice is quite straightforward and simple. Although the usage cost remains unchanged, it requires a static IP setup by the service provider and involves monthly recurring costs. The static IP is preferred for its simplicity in designing a system; however, its overhead may be impractical. The other choice is to use a dynamic IP assigned through a Dynamic Host Configuration Protocol (DHCP) server of the GSM provider for every connection established. However, this IP needs to be known by any client requesting an access to the embedded server. One solution is to broadcast this IP to a dummy FTP server (where the bulky static information such as image data is also kept). The FTP server is a dummy server and does not require regular software updates or maintenance. The folder structure of the FTP server is shown in Fig. 2.

A script on the embedded device is configured to update its IP address on the FTP server in Hypertext Meta-Language as an index.htm file, under a folder uniquely named by its hostname. This script simply parses the current IP for that embedded device and sends an htm file with the IP information of the embedded device to the FTP server. Once this file is in place, a direct connection can be established with the desired

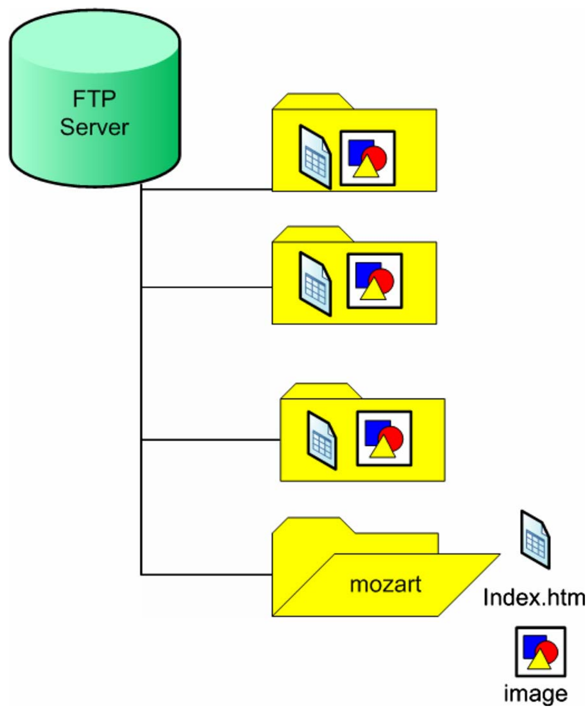

Fig. 2. Folder structure of the FTP server.

embedded device by a simple query. An example embedded system, named *mozart* (Fig. 2), can be queried from the FTP server by a simple command. The web browser processes the (index.htm) file in the specified folder as default; therefore, a file name is not needed for referencing.

This process is illustrated in Fig. 3. The DHCP approach is more flexible and works better compared with the static approach as a cost-effective solution, despite the necessity for a script running on the embedded server, one-time broadcasting its IP to the FTP server.

The hypertext file placed on the FTP server by the embedded system and queried by the client is shown in Fig. 4.

With this mechanism in place, the embedded system updates it IP information on the FTP server upon every reboot, which causes an IP refresh from the GSM service provider.

B. Data Management in the System

The Internet server is used to decrease the management costs by sending all the pictures (logo, picture, bar graphics, etc.) to the client through a server on the Internet. Text data such as coordinates, temperature, and altitude are served from the embedded system. If bulky data are going to be sent, the embedded module is set to send the image only once via GPRS and placed on an FTP server. This approach eliminates the transfer of large data through GPRS more than once, thus reducing the transfer costs, particularly if more than one client is involved or multiple requests to the same data are needed, as shown in Fig. 5.

A user interface, which is brought up upon establishing a direct connection, has links to the Common Gateway Interface (CGI) and Bourne Again Shell (BASH) script files executed on the embedded system. The code is compiled into the CGI format to be installed in the embedded board through

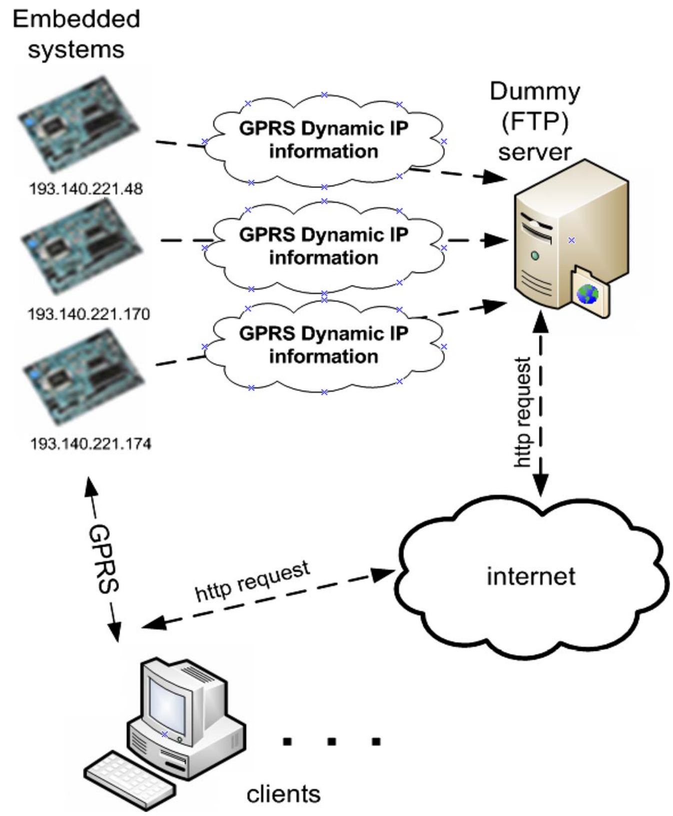

Fig. 3. IP address lookup for the stored IP.

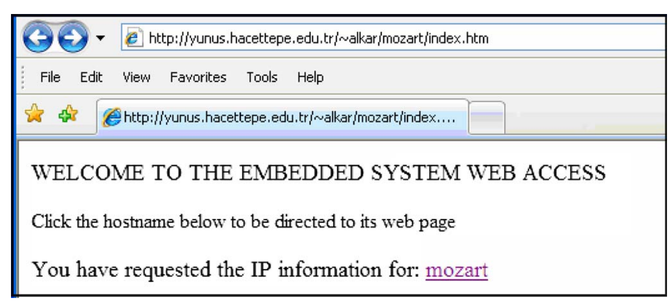

Fig. 4. Index.htm on the FTP server as viewed by the client.

a cross-compiler platform [22]. BASH scripts are directly triggered by the applications.

C. Hardware

The embedded system used in this work is an X86-based standalone unit with four serial ports and a parallel port with 16-MB onboard removable Flash memory, as shown in Fig. 6.

One of the serial ports is used in the application design stage for debugging purposes, and this port is designed to host more devices with a multiplexer unit. The other serial ports used by the modules[1] are used to test the system functionality. The acquisition units on the device can be varied with no limitation

[1]A GM862 GPRS modem is used in the second serial port. A UV-40 GPS receiver is used in the third module. Port 4 is dedicated for the camera module, where a CAM-VGA100 is used. A parallel port is used to connect a DS1620 temperature sensor.

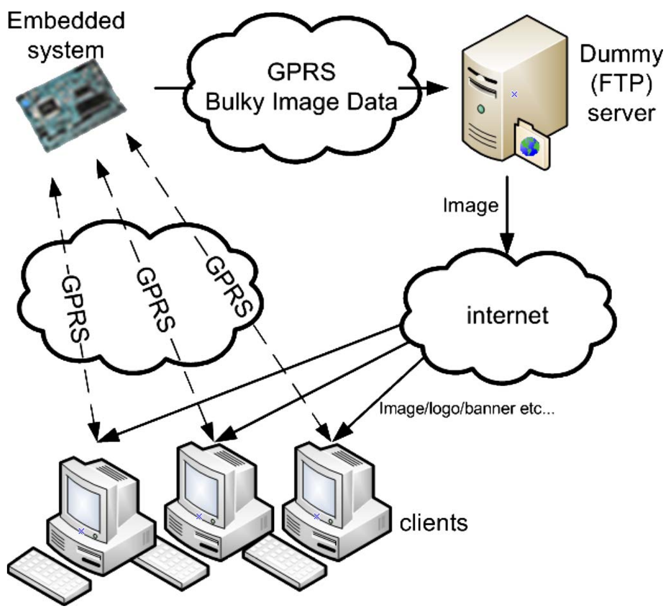

Fig. 5. Data management in the proposed system.

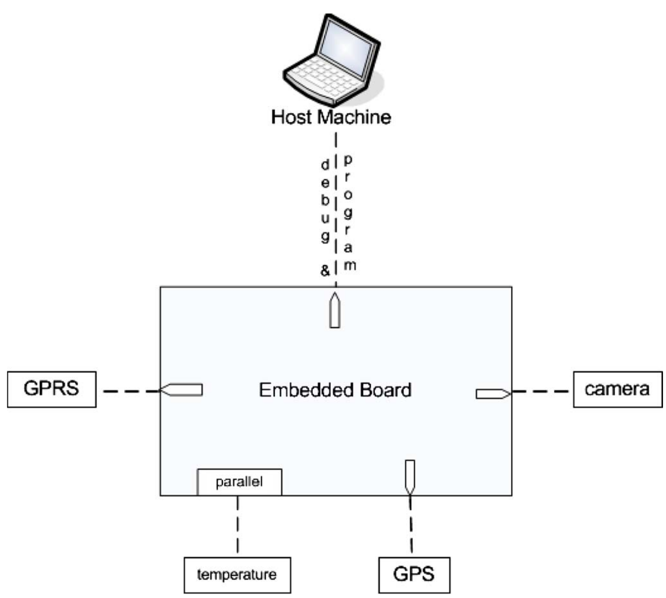

Fig. 6. Block diagram of the embedded system with sample devices attached.

on their functionality and can be added by using appropriate interfaces.

D. Software and Operating System Choice

The Linux 2.4 kernel series [23] with TCP/IP stack included has been chosen as the operating system for the embedded board. Only the bare minimum is installed, including the basics such as console tty, serial ports, kernel side of the PPPD, and support for memory and math emulation. The running kernel is around 1 MB of code built into a Flash memory. A scaled-down version of Linux has been used to reduce the memory footprint and the complexity.

The software running on the embedded system at the highest level is named the manager code, which will be explained in Section III with a sample implementation. In the design, the manager code controls the execution of other applications and is triggered once all the components of the operating system are up and running. The flowchart of this code is given in Fig. 7, representing a sample operation of the GPS unit that checks the execution if the speed limit has been exceeded.

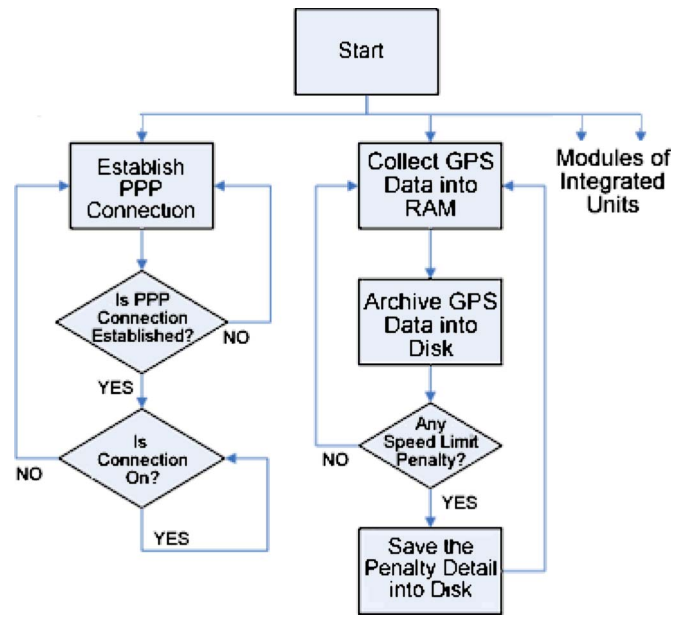

Fig. 7. Manager code operational principles.

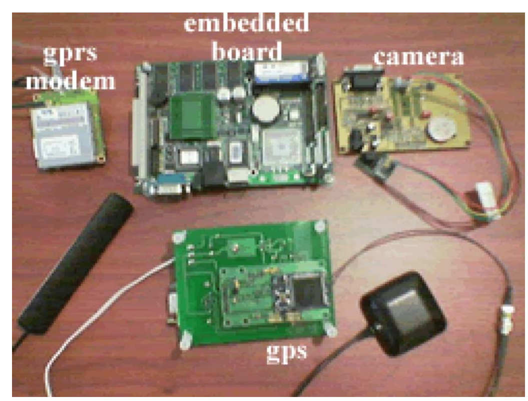

Fig. 8. Components of the embedded system.

The periodic operations and routine tasks are organized by a manager code. If a new data-access application is considered in future developments, its program can easily be added to the manager code as a periodic operation.

III. Sample Application

A camera, a temperature sensor, and a GPS are integrated into the embedded board to form a sample application, as shown in Fig. 8. These units and their interaction with the embedded board are briefly explained in the following discussion. In addition, the time delays at each operation stage are discussed to demonstrate the effectiveness of the proposed method.

A time delay of 0.57 s, on average, is needed for any control command to be sent. This delay is related to the GPRS service of the GSM service provider.

TABLE I
TOTAL CONNECTION TIME TO RECEIVE A 16-kB PICTURE THROUGH AN FTP SERVER AND DIRECT ACCESS

# of clients at the same time	Σ Duration (sec) to receive pictures	
	FTP server	Direct Access
1	21.77	21.77
2	21.77	40.14
3	21.77	58.51
4	21.77	76.88

For this sample application, a complementary metal–oxide–semiconductor camera with a built-in JPEG codec controller chip has been chosen. The camera acquires bulk image data; therefore, it is a good module to demonstrate the effectiveness of the system. It compresses and transfers the image from the camera to the serial port. The communication with the camera is established over an RS232 communication protocol using an asynchronous package transfer method. Before taking a snapshot, the camera is synchronized by sending an appropriate number of synch data packages. After the synchronization, both the embedded board and the camera wait until they receive an acknowledgement from the other side before sending another request or data. This protocol is executed in an average of 3.4 s for each picture, which can be considered as an adequate rate for most applications. Here, the bottleneck is the camera; hence, the speed of data transfer can further be improved by using a camera with a faster sampling rate. The client initiates the camera control script, which eventually takes a snapshot. The embedded board receives the data from the camera port then stores them into the Flash memory externally added onto the embedded unit. The available data storage in 16-MB Flash memory is 6.6 MB, which is suitable for over 420 pictures. The picture is uploaded (a 16-kB JPEG picture is transferred in an average of 17.8 s) to a dummy FTP server, as described in Section II-A. The server on the Internet is not maintained and only used for storage space. Since, for our application, a history is not required to be kept, the client(s) accessing the picture download(s) the most recent snapshot from the FTP server. All the queries to visualize the current picture are automatically relayed to the FTP server instead of the embedded module, thus reducing the operational costs. The duration comparison of using a dummy server with respect to direct access is shown in Table I.

The scenario of serving a single picture to a maximum of four clients is shown in Fig. 9. The x-axis represents the number of client accesses to a picture. For the camera application, the client can take a snapshot and visualize the picture on the screen (an average of $0.57 + 3.4 + 17.8 = 21.77$ s for a 16-kB picture) in less than 22 s, including the time delay of command execution, camera operation, and picture upload duration. Although this may seem like a large delay, it may be improved with a faster and more expensive camera. The transfer of text data takes an average of less than 1 s (∼1 kB). The solid line in Fig. 9 illustrates the case where the client is served by the embedded system and an FTP server. As the number of clients

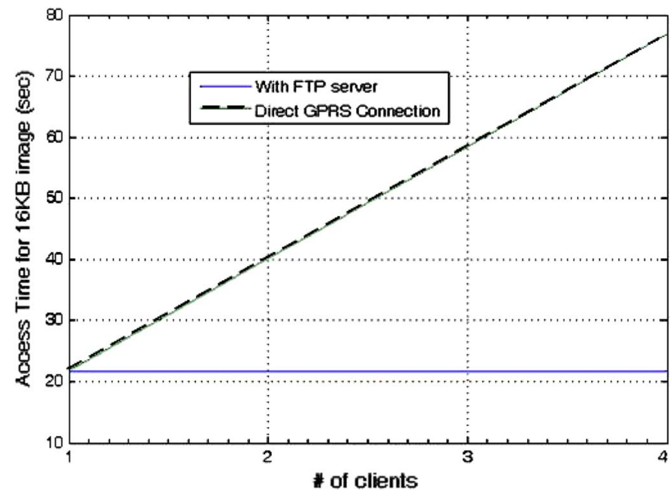

Fig. 9. Number of clients with respect to duration for picture transfer.

Fig. 10. Dispatcher application snapshot.

increases, the data to be served remain constant with respect to the clients for the proposed system since the bulk of data is relayed through the FTP server.

If the FTP server is not used, then for each client, the data will directly be served through GPRS repeatedly, causing a linear increase in the response time requirements. In this case, the connection time is proportional to the amount of data to be transferred through GPRS. With the FTP-server-supported connection, a constant cost is obtained with respect to the number of clients, and the usage cost of the system is reduced.

An actual snapshot taken by the embedded system is shown in Fig. 10. In this application, the camera visually shows the location to the client. For a dispatcher application, for instance, the client can visualize the environmental or cargo conditions.

The snapshot script used in this application is given in Fig. 11.

The routing information of the vehicle can be collected for the analysis of speed violations, altitudes traveled for sensitive

```
Snapshot()

Connect Embedded_Board_to_Camera

  Send Synch packages to Synchronize

Wait until response received from camera

Take_Snapshot

  Camera_Execute_Snapshot

Receive_Snapshot

  Store_into_Flash

  Upload to FTP site

Close_Connection
```

Fig. 11. Snapshot pseudocode.

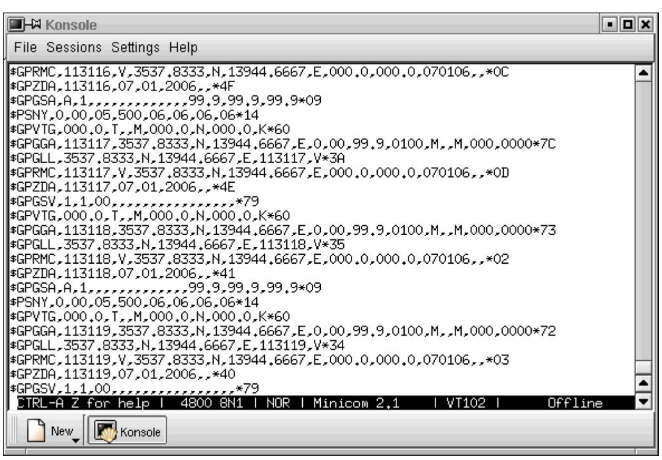

Fig. 12. Raw GPS data sample.

cargo, etc. The GPS module used in the application is an OEM GPS UV40, serving the NMEA 0183 [24] format raw data with a simplex communication protocol operating on one of the serial ports. The program transfers the selected GPS data to the memory after compiling a bulk of raw data. Useful information is parsed from these raw data within the embedded server by detecting the starting points of the NMEA data as reference points while reading into the serial buffer. A raw GPS data sample is shown in Fig. 12.

The embedded board starts a periodic operation of acquiring the raw data from the GPS module just after the boot time of the embedded Linux. An admin script is responsible for periodically executing the GPS program and sending the qualified GPS data to the Flash memory. In order not to exceed the Flash memory limit, the newest GPS data are exchanged with the oldest data using the memory as a FIFO buffer. This provides up-to-date GPS data to be available at the FIFO upon logging into the system. This can be critical in some real-time applications.

Information such as the current altitude and location can be logged. The system can be set to archive the past data associated with a certain date if logging is necessary. The altitude measurements are accurate up to 20 m, which is also within the accuracy limits for a typical GPS. The sampled raw GPS data are processed, as shown in Table II. The ernestGPGGA information is parsed from the raw data and stored in a file to be further processed by the CGI scripts.

The system can be set to visually track the current location of the embedded system on a map. In Fig. 13, a snapshot displays the location information for the vehicle on a map using the processed data obtained from the GPS. An icon that represents a vehicle is inserted into the location using basic frames in html, utilizing the latest coordinate information from the GPS data on the map image retrieved from the FTP server. The resolution can be increased by increasing the number of frames used in html. The GPS accuracy of the measurements is less than 15 m, which is typical for a civilian GPS module. The error in accuracy is known to be due to atmospheric effects, such as the ionosphere or multipath effects.

Due to the necessity of using pure html code due to web server limitation in embedded systems, the generation of the pages through CGI scripts is simple yet informative.

In addition, a temperature measurement integrated circuit with a very low cost temperature measurement chip (namely, DS1620) is used to collect ambient temperature in certain time intervals at an accuracy of 0.5 °C. This chip is attached to the parallel port of the embedded board. A daemon is initiated at boot time to sample and display the temperature every 30 s for a time interval of 15 min.

The output graph generated for temperature is shown in Fig. 14. This enables the user on the client to constantly monitor the temperature. Fig. 15 shows a snapshot displaying the altitude information.

The embedded system is accessible via a web server built into the device. This is nothing more than a CGI, which accepts data, processes them, and returns an answer. A small web server, named Boa Server, which is particularly targeted for embedded systems, is used in the embedded system, which keeps the memory requirement minimum. Unlike most other web servers, Boa does not use multiple threads or processes to serve multiple clients at the same time but uses smart usage of the select function. This reduces kernel scheduling activities.

The sample system is low cost, including the main module, the GPS, the camera, and the temperature sensor. It was necessary to design and implement some of the necessary interfaces for the RS232 communication with the main module.

IV. Conclusion

In this application, a low-cost, Internet-based data-acquisition and control system has been designed and implemented that should find interest from researchers. The

TABLE II
SAMPLE RAW GPS DATA AND CORRESPONDING ALTITUDE AND LOCATION INFORMATION

$GPGGA,151732.750,3952.1503,N,03244.1166,E,1,03,7.7,1172.4,M,,M,,0000*7A	
Type	$GPGGA
UTC time	151732
Latitude	3952.1503,N
Longitude	3244.1166,E
Number of Satellites Connected	03
Altitude	1172.4,M

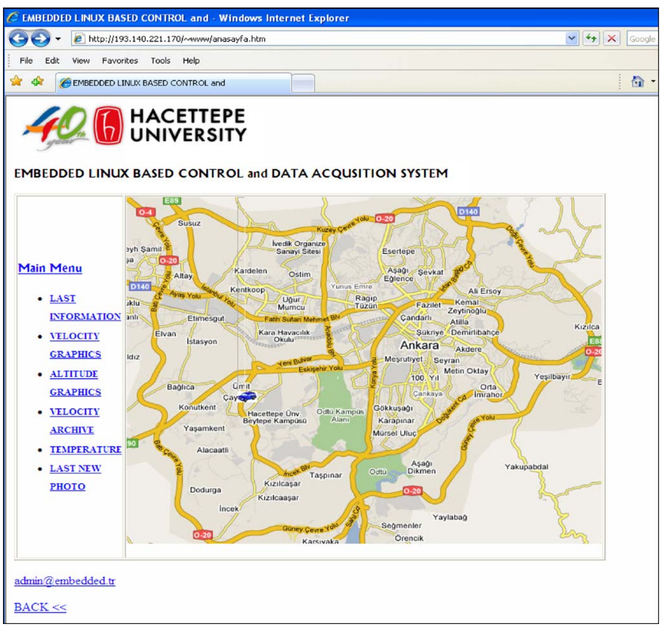

Fig. 13. Map screen for the client indicating the location of the vehicle on the map.

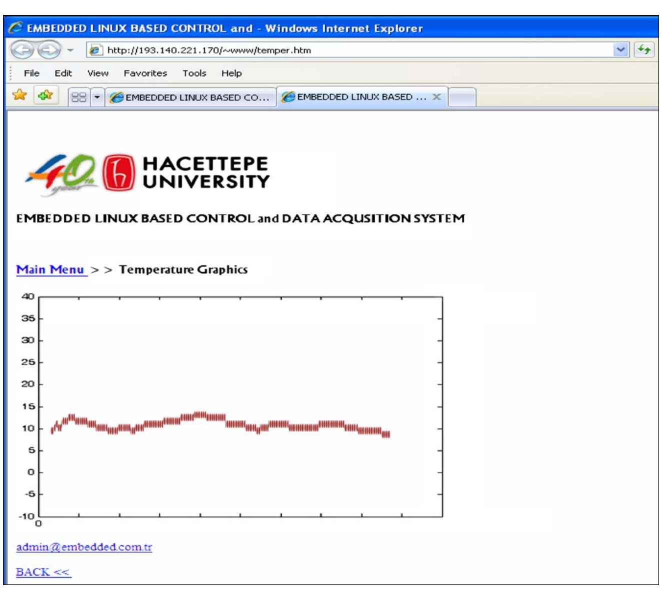

Fig. 14. Corresponding speed/time graph.

application possibilities are virtually unlimited by attaching modules with appropriate interfaces, although the usage of the system is demonstrated with only a few sample devices.

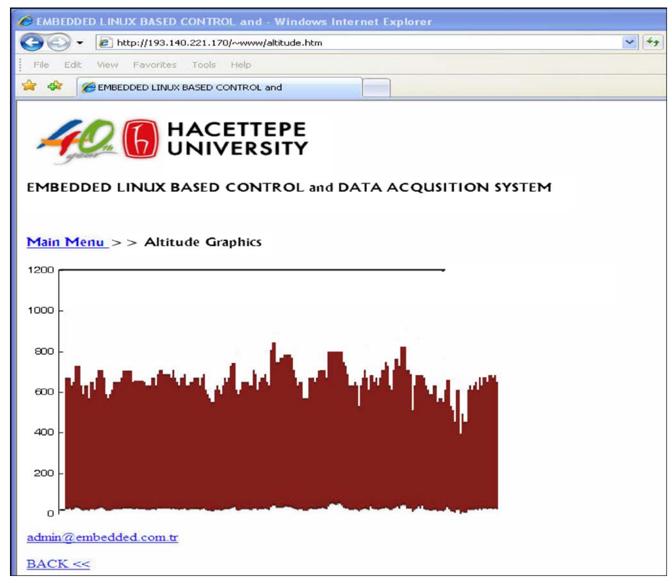

Fig. 15. Altitude graphics from the embedded system.

Compared with other applications, this system has advantages in terms of allowing direct bidirectional communication and reducing overhead, which can be vitally important for some real-time applications. The operational costs have been reduced by relinquishing the storage of large data to an FTP server on the Internet.

The system is designed to support both static and dynamic IPs. A method to distribute the IP information has been developed. This cost-minimization effort is a big concern for mobile systems using wireless communication methods and has not been discussed before.

The overall cost advantage of the system in terms of the components used makes it an attractive choice for data-acquisition applications.

The power demand of the device is still in the process of being improved by putting the attached devices into sleep mode at times when they are not in use to conserve power.

REFERENCES

[1] C. E. Lin, C.-W. Hsu, Y.-S. Lee, and C. C. Li, "Verification of unmanned air vehicle flight control and surveillance using mobile communication," *J. Aerosp. Comput. Inf. Commun.*, vol. 1, no. 4, pp. 189–197, Apr. 2004.

[2] K. Jacker and J. Mckinney, "TkDAS—A data acquisition system using RTLinux, COMEDI, and Tcl/Tk," in *Proc. Third Real-Time Linux Workshop*, 2001. [Online]. Available: The Real Time Linux Foundation: http://www.realtimelinuxfoundation.org/events/rtlws-2001/papers.html

[3] E. Siever, A. Weber, S. Figgins, and R. Love, *Linux in a Nutshell*. Sebastopol, CA: O'Reilly, 2005.
[4] Q. Zhou, W. Wu, and Y. Ma, "The embedded data acquisition system for Mössbauer spectrum," in *Proc. Third Real-Time Linux Workshop Embedded Linux Expo Conf. Real-Time Embedded Comput. Conf.*, Milan, Italy, Nov. 2001, pp. 26–29.
[5] J. E. Marca, C. R. Rindt, M. Mcnally, and S. T. Doherty, "A GPS enhanced in-vehicle extensible data collection unit," Inst. Transp. Studies, Univ. California, Irvine, CA, Uci-Its-As-Wp-00-9, 2000.
[6] W. Kattanek, A. Schreiber, and M. Götze, "A flexible and cost-effective open system platform for smart wireless communication devices," in *Proc. ISCE*, 2002.
[7] J. E. Marca, C. R. Rindt, and M. G. Mcnally, "The tracer data collection system: Implementation and operational experience," Inst. Transp. Studies, Univ. California, Irvine, CA, Uci-Its-As-Wp-02-2, 2002.
[8] E. Bekiroglu and N. Daldal, "Remote control of an ultrasonic motor by using a GSM mobile phone," *Sens. Actuators A, Phys.*, vol. 120, no. 2, pp. 536–542, May 17, 2005.
[9] C. E. Lin and C.-C. Li, "A real-time GPRS surveillance system using the embedded system," *J. Aerosp. Comput. Inf. Commun.*, vol. 1, no. 1, pp. 44–59, Jan. 2004.
[10] C. Xiaorong, S. Zhan, and G. Zhenhua, "Research on remote data acquisition system based on GPRS," in *Proc. 8th ICEMI*, 2007, pp. 2-20–2-23.
[11] M. A. Al-Taee, O. B. Khader, and N. A. Al-Saber, "Remote monitoring of vehicle diagnostics and location using a smart box with Global Positioning System and General Packet Radio Service," in *Proc. IEEE/ACS AICCSA*, May 13–16, 2007, pp. 385–388.
[12] C. Zhang, J. Ge, H. Yu, and X. Zhang, "ET gravimeter data collecting system based on GPRS," in *Proc. 8th ICEMI*, Jul. 18–Aug. 16, 2007, pp. 2-86–2-92.
[13] A. Sang, H. Lin, and C. E. Y. Z. Goua, "Wireless Internet-based measurement architecture for air quality monitoring," in *Proc. 21st IEEE IMTC*, May 18–20, 2004, vol. 3, pp. 1901–1906.
[14] G. Zhenyu and J. C. Moulder, "An Internet based telemedicine system," in *Proc. IEEE EMBS Int. Conf. Inf. Technol. Appl. Biomed.*, 2000, pp. 99–103.
[15] J. Dong and H. H. Zhu, "Mobile ECG detector through GPRS/Internet," in *Proc. 17th IEEE Symp. CBMS*, Jun. 24–25, 2004, pp. 485–489.
[16] P. Wang, J.-G. Wang, X.-B. Shi, and W. He, "The research of telemedicine system based on embedded computer," in *Proc. 27th IEEE Annu. Conf. Eng. Med. Biol.*, Shanghai, China, Sep. 1–4, 2005, pp. 114–117.
[17] C. E. Lin, C.-C. Li, A.-S. Hou, and C.-C. Wu, "A real-time remote control architecture using mobile communication," *IEEE Trans. Instrum. Meas.*, vol. 52, no. 4, pp. 997–1003, Aug. 2003.
[18] T. Motylewski, "The industrial data-acquisition system with embedded Rt-Linux and network server technology," in *Proc. Third Real-Time Linux Workshop*, 2001. [Online]. Available: The Real Time Linux Foundation: http://www.realtimelinuxfoundation.org/events/rtlws-2001/papers.html
[19] S. H. Yang, L. S. Tan, and X. Chen, "Requirements specification and architecture design for Internet-based control systems," in *Proc. Int. Comput. Softw. Appl. Conf., Dev. Redev.*, 2002, pp. 75–80.
[20] C. Bettstetter, H.-J. Vögel, and J. Eberspächer, "GSM phase 2+ General Packet Radio Service GPRS: Architecture, protocols, and air interface," *IEEE Commun. Surveys Tuts.*, vol. 2, no. 4, pp. 2–14, Third Quarter 1999.
[21] RTOS Evaluation Project, "What makes a good RTOS," *Dedicated Systems Experts*, 2001. [Online]. Available: http://www.dedicated-systems.com
[22] K. Yaghmour, *Building Embedded Linux Systems*. Sebastopol, CA: O'Reilly, 2003.
[23] C. Hallinan, *Embedded Linux Primer: A Practical Real-World Approach*. Englewood Cliffs, NJ: Prentice–Hall, 2006.
[24] NMEA, *NMEA 0183 Standard*. [Online]. Available: http://www.nmea.org

Ali Ziya Alkar (S'93–M'95) received the B.S. degree from the Electrical and Electronics Engineering Department, Hacettepe University, Ankara, Turkey, in 1988. He received the M.S. and the Ph.D. degrees from the Electrical and Computer Engineering Department, University of Colorado, Boulder, in 1991 and 1995, respectively.

He is currently an Assistant Professor with the Department of Electrical and Electronics Engineering, Hacettepe University, supervising and completing several government-funded research projects. His main interests are VLSI, microprocessor architecture design, access control systems, and embedded systems.

Mehmet Atif Karaca received the B.S. and M.S. degrees from the Electrical and Electronics Department, Hacettepe University, Ankara, Turkey, in 2003 and 2006, respectively.

He is currently with Sistemim Electronics Design and Consulting Company, Ankara. His research interests include biometric automation and the design of data acquisition and embedded systems.