CHAPTER V

CONCLUSION

The conclusion about all the done work so far will be presented here. In other words, it is an evaluation of the proposed scheme and on whether the implementation method satisfies the research's requirements. Discussion about to which applications that the proposed solution might be suitable to be applied and what the next researches on this matter should focus on, to make the solution more complete, will be given. The amount of money spent to complete this research is also estimated.

To address the issues mentioned above, this chapter is presented first with the summary of results which have been gained. The second part is the possible applications in which the proposed scheme can work. Next are the suggestions on future works and the last part of content is the cost analysis.

5.1 Summary

In this research, we have proposed a way to implement SCADA systems by means of Ethernet and Internet networks. The test platform is based on a web-based application and a simulation scheme. Visual C# was used to write the web-based interface, which is a website, for the entire system and Matlab was used to simulate the system devices.

- ***Chapter 1*** is the introduction about SCADA technology and the Internet-based solution in comparison with other types of communication networks. Other matters including objectives, research scope, methodology and contribution of the research was also given.

- ***Chapter 2*** is a summary on the literature review work. A number of papers have been read to find out the existing problems. A short description of what the proposed scheme might bring was also mentioned.

- ***Chapter 3*** presented the features of the proposed scheme and the generally tentative architecture of the system. All the basic technologies to be used to build the system and the overall understandings about these technologies were also mentioned.

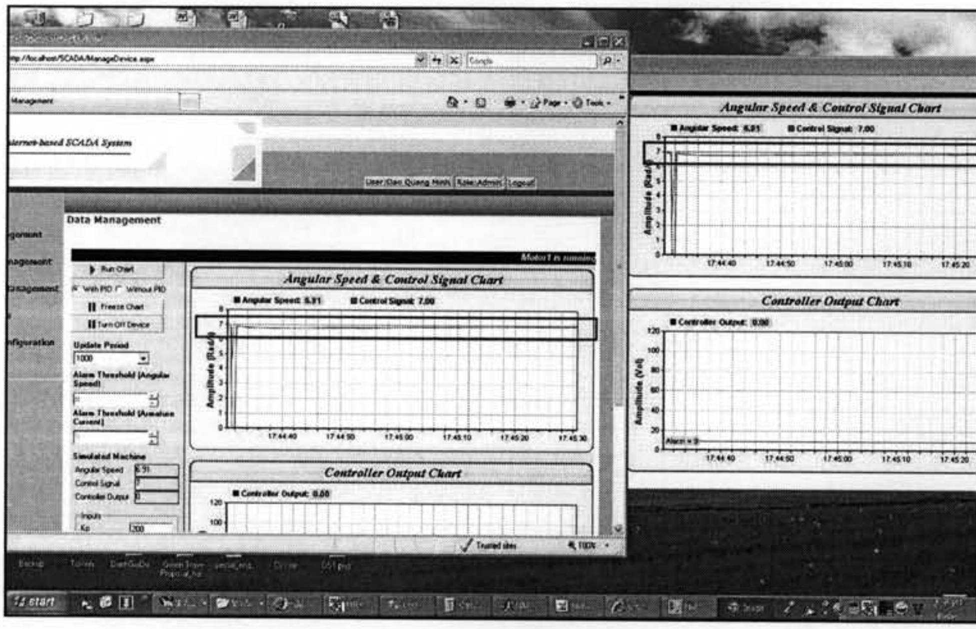


Figure 5.1 DS and SCADA interface at the same time operating

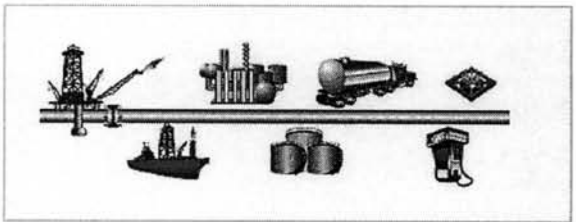
- *Chapter 4*, which contains the main contents of the research, talked about the complete implementation of the proposed scheme. Details about system hardwares and software tools to be used were presented. Here, deep details about the operational principles which correspond to each of the system's features in real operation were written. Matlab was used to simulate the operation of two DC motors which are to be used as the system's devices. Matlab code was used together with VC# code to make the so-called DS modules. The module reflects exactly the operation of a DC motor with an IP address already attached. There are two operation schemes for the simulated motors provided which are under PID controller and free operation without the PID controller. Communications from and to this module are done over both the Ethernet and Internet networks under the TCP/IP protocol standard. The system is built as a web-based solution with the presence of a server computer which serves as both the application server and DB server. Details about how to connect and work with the system DB were given. Detailed data in each DB table were also presented by Figures for convenient readings. In the next part in chapter 4, details and screen-shots of the operation of each system function were given in an appropriate order so that readers can understand how to operate the system. The most outstanding functionality of the system is the real-time data monitoring illustrated in the Figure 5.1 with both the DS interface and the SCADA pages showing the same data progress.

The last part of content in this chapter is a discussion about the problems faced during the testing process together with tentative solutions.

1.12 Recommended Applications

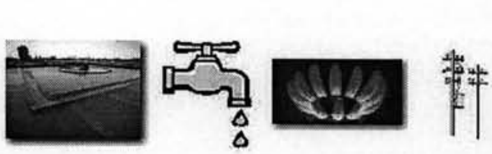
Like other SCADA solutions do, the Internet-based solution provides the ability to remotely collect and monitor process data as well as to control process devices in most applications in various industries. Some possible applications are listed below:

1. **Fuel Tracking Systems**



Fuel pipelines often require the ability to monitor the fuel flow. Leakage must be detected in time for maintenance. Fast responses are not strictly required.

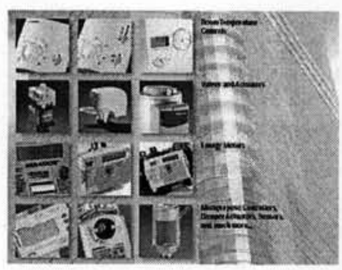
2. **Irrigation, Weather Forecast Systems**



Irrigation or Forecast systems also do not require fast reactions or controllability but monitoring ability. Therefore, Internet-based SCADA solution might be best suitable

3. **General Applications:**

- Water Leakage Detecting system.
- Pipeline Management system.
- Water and Waste water Treatment system.
- Building Automation.
- Alarm systems (Flood, Tsunami, etc.)



There are various applications in which the Internet-based solution can be applied. However, for those applications which require very fast response or urgent reactions such as Power Failure Detection, Fire Alarm System, etc. or Emergency Reaction Systems, this solution is not recommended for the reason that it may not give an in-time response if the Internet is congested.

1.13 Recommendations for Future Work

5.3.1 Improvements on the testing system

The testing system was built based on the software simulation basis. Due to the time limit, the research could not cover all aspects of a SCADA implementation. In this part of the research, restrictions and the way to improve them will be discussed:

1. **With the DS Module**

- *Problem Addressing:*

By looking again at the DS interface (Figure 5.2), we can note the points at which the angular speed is 0 (points which are circled). The value of 0 appears whenever there's a command sent to the DS module by system users. In fact, it should not have happened. The simulated motor should have run smoothly and the smooth transient among speed values of the simulated motor is required. With the current operation as shown on Figure 5.2, we assumed that the simulated motor is turned off before it can react to a new command.

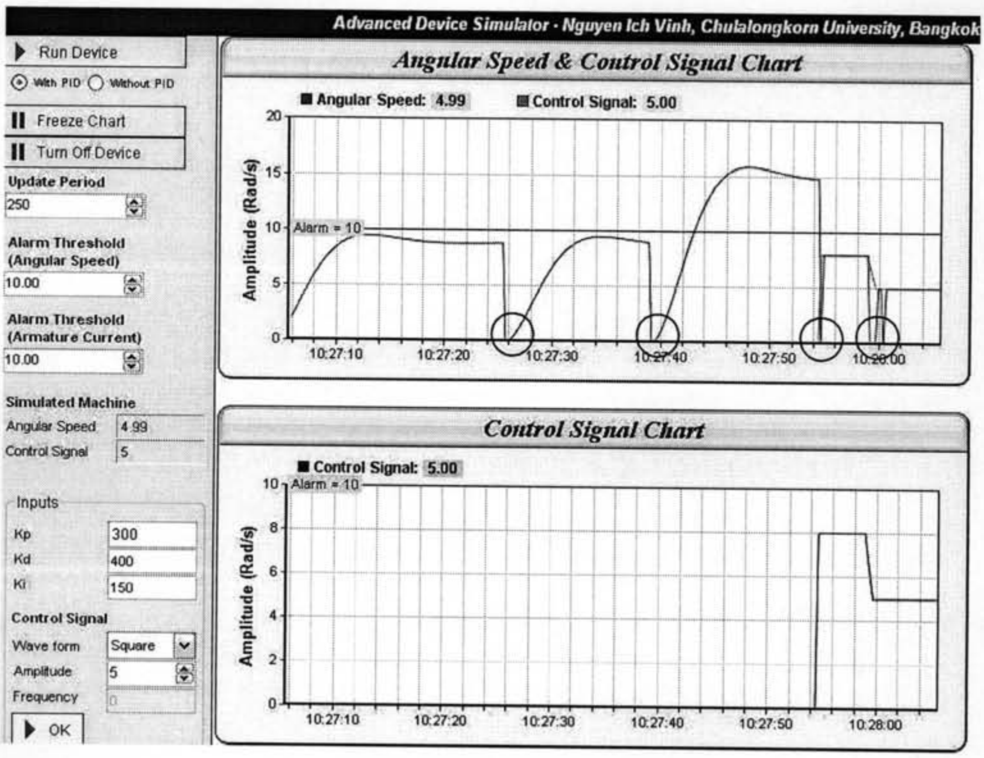


Figure 5.2 DS Interface with 0 points

- *Solution:*

To avoid this error and to better reflect the real operation of the DC motor, more efforts must be put in programming the Matlab files in which we can:

1. Memorize the latest value of the motor's angular speed
2. Take this value as the initial state of the motor for the next time the Matlab files run upon receiving a new command.

By doing this way, users can get rid of the unwanted 0 values each time the DS module reacts to a new command. The graphs representing the angular speed and the armature current of the simulated motor will run continuously. There's no assumption that users have to turn off the motor before giving new commands to the DS module.

2. System Security Issue

- *Problem Addressing:*

The testing system has well presented four security reasons. Three of the four, which are User Authentication, IP Approval and Router Configuration, are for the purpose of preventing the unwanted connections to the system. The User Permission tool is to restrict users' control on the system. These provided tools, however, do not guarantee that the system's data can travel safely on the Internet. In case of data trapped, outsiders can translate these data and understand what is going on in our system. This is what we do not want to happen

- *Solution:*

In order to protect the system data, system data need to be encrypted before being sent and at the receiver; these data again need to be decrypted into an understandable format. We will need to program for the ActiveX control at the client side to encrypt requests before sending them to the server then to the DS module. The ActiveX control at the server computer, upon receiving the data from the DS module, has to encrypt these data and send them in the response to the client's request. The ActiveX control at the client side, once received the response, again it has to decrypt the encrypted data and display under the graphical and numerical forms on the client's SCADA pages.

One of the most common encryption algorithms used in Internet communication is MD5. MD5, which stands for Message-Digest 5, processes a variable-length message into a fixed-length output of 128 bits. The input message is broken up into chunks of 512-bit blocks; the message is padded so that its length is divisible by 512. The padding works as follows: first a single bit, 1, is appended to the end of the message. Zeros as are then added to build up the length of the message up to 64 bits fewer than a multiple of 512. The remaining bits are filled up with a 64-bit integer representing the length of the original message.

The main MD5 algorithm operates on a 128-bit state, divided into four 32-bit words, denoted by *A*, *B*, *C* and *D*. These are initialized to certain fixed constants. The main algorithm then operates on each 512-bit message block in turn, each block modifying the state. The processing of a message block consists of four similar stages, termed *rounds*; each round is composed of 16 similar operations based on a non-linear function, modular addition, and left rotation. Figure 1 illustrates one operation within a round. There are four possible non-linear functions called *F, G, H and I*; each one is used in each round.

$$F(X,Y,Z) = (X \wedge Y) \vee (\neg X \wedge Z)$$
$$G(X,Y,Z) = (X \wedge Z) \vee (Y \wedge \neg Z)$$
$$H(X,Y,Z) = X \oplus Y \oplus Z$$
$$I(X,Y,Z) = Y \oplus (X \vee \neg Z)$$

$\oplus$, $\wedge$, $\vee$, $\neg$ denote the XOR, AND, OR and NOT operations respectively

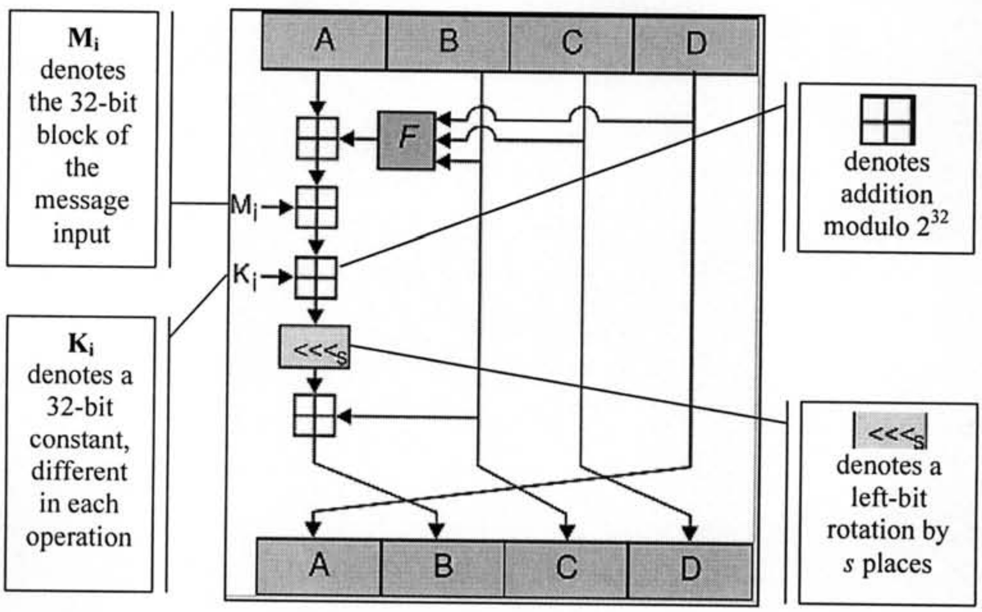


Figure 5.3 MD5 Operation

Figure 5.3 describes one MD5 operation within a round which the presence of the function *F*.

The 128-bit (16-byte) MD5 hashes (also termed *message digests*) are typically represented as a sequence of 32 hexadecimal digits. For example, the hash of the zero-length string is:

MD5("") = d41d8cd98f00b204e9800998ecf8427e

MD5 is considered to be a good algorithm since its operation will give a completely new hash even with a small change in the input message.

Performing the MD5 operation is the task of the ActiveX control as mentioned above.

3. Efficiency of Internet Communication

- *Problem Addressing:*

As mentioned, Internet communication depends much on the traffic of the network. It also depends on the bandwidth at both the server and client sides. In this research, ADSL technology was used at both server and client side. 25 ms was used as the interval between two requests to server by client. We could achieve the running data graphs in real time and the delay time for each data value to be displayed was at 47-50 ms, which is small enough to consider the system a as real-time system. However, from Figure 5.4 and as mentioned in the last part of Chapter 4, we could note that the data graphs on client SCADA pages are not as smooth as the ones on the DS interface. What we are expecting is that the SCADA graphs can smoothly and accurately reflect what is happening with the DS module

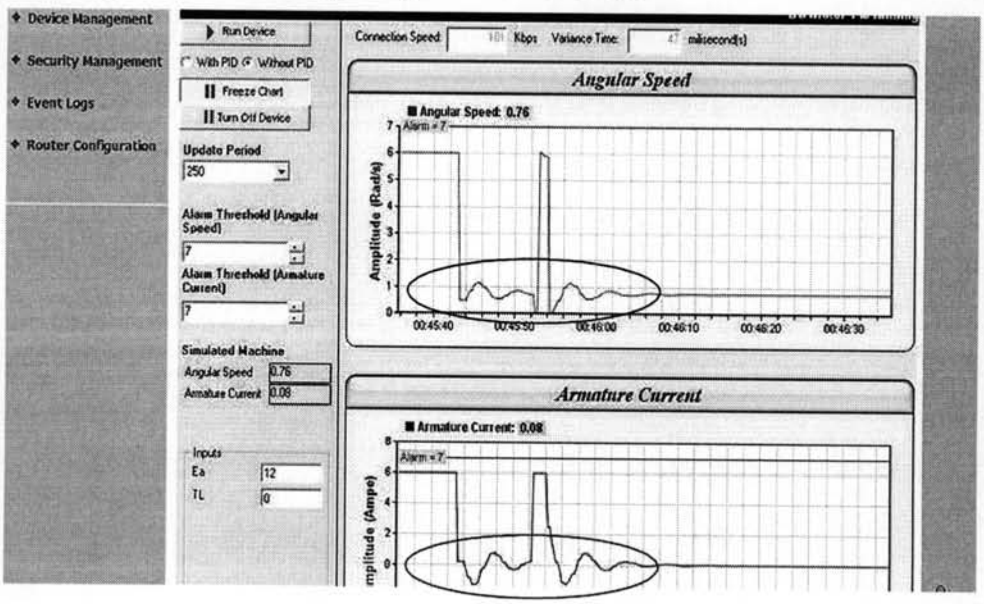


Figure 5.4 Un-smooth data graphs

- *Solution:*

To improve the quality of the data graphs, we can have several ways:

1. Reduce the data exchange interval

25 ms was chosen as a rule-of thumb, it is small enough to ensure accurate data retrieval from server and a real-time manner. We can reduce this interval to less than

25 ms, but we might be at the risk of losing the first data value if the data of the second request comes to the server before the response to the first request is received by the client. In that case, the data value that the client actually receives is the response to the second request, and the data value of the first response has been lost. With this solution, the system designer needs to consider the quality of the quality of the Internet network at both the server and client sides, and also the point of time to use the system, to see whether it's possible to reduce the interval.

2. **Increase the data exchange interval**

The server application is a website; all communications between clients and server conform to http protocol. Every 25 ms, a request is sent to the server and thereafter, there is a response sent back to the client, the HTTP headers and footers will be added to make up the HTTP frames. These frames are the actual data exchanged between server and client.

We can increase the smoothness of the SCADA graphs by defining the interval to be longer than 25 ms. By doing so, each request from client will result in a larger amount of data in the response. We then have a smoother display of data on client SCADA pages and we can also reduce the huge number of HTTP headers and footers.

However, increasing the data exchange interval means that we are reducing the real-time factor of the system. Again, which value of the interval needed must be considered carefully in programming the server application software.

3. **Invest more on the allowed bandwidths at both server and client sides**

This option is not recommended in the testing system. Demand for larger bandwidth results in more investment. Nevertheless, higher bandwidth allows the system designer to optimally minimize the data exchange time while still ensuring the accurate data retrieval. With higher communication bandwidth, we can also ignore the amount of HTTP headers and footers, which reduce the programming efforts.

5.3.2 Recommendations on newly-built systems

1. Hardware Components

- *The server computer* can be upgraded depending on the requirements of storage capacity and communication ability of the system. An example of an industrial server is shown in Figure 5.5.

This kind of computers can work as a stand-alone machine in the hazardous conditions.

They also provide a faster data processing speed; increase the DB capacity and the communication ability of the entire system.

Figure 5.5 Industrial Server

- *Ethernet board*

Gateways with higher data security and expanded communication ability can be used instead of Ethernet boards for the data collection purpose.

A sample of a gateway, manufactured by DevicesWorld, is presented in Table 5.1.

Table 5.1 Gateway

		- Local Data Logging - Event - driven Dial-up - On Demand Connection - Password Protection & Data Encryption - Extended Digital I/O and Analogue I/O
Gateway	**I/O Expansion**	**Advanced Features**

Gateways not only act as the data acquisition part, but they also provide extra features which increase the communication ability as well the security of the system.

2. Software Components

o **The existing interface** in this research can be customized to be suited with each specific application. This interface can be developed to monitor large-scale processes in unlimited areas. Illustration is presented in Figures 5.6 and 5.7.

(Referred from www.devicesworld.net/)

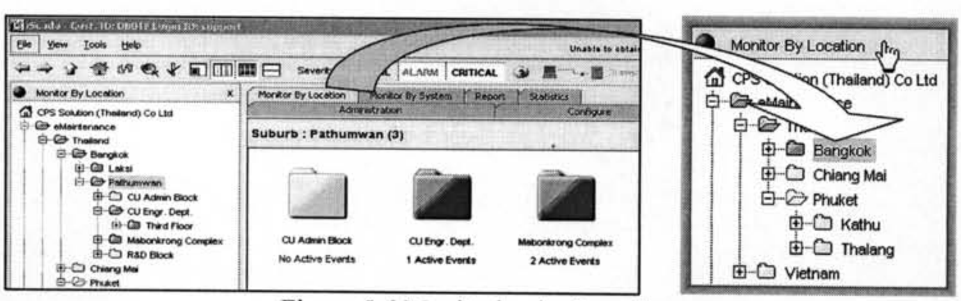


Figure 5.6 Monitoring by Location

Users can take control on the wanted process based on the location or on each specific running system in the whole process

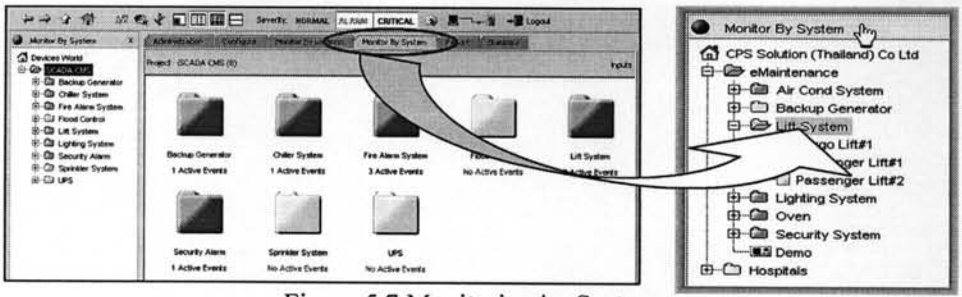


Figure 5.7 Monitoring by System

We can also add the automatic mobile alarm to the system's functionalities. The system operator is equipped with a mobile phone or a Personal Digital Assistant (PDA). The system will be configured to automatically send an alarm or email to the operator in case of unwanted events.

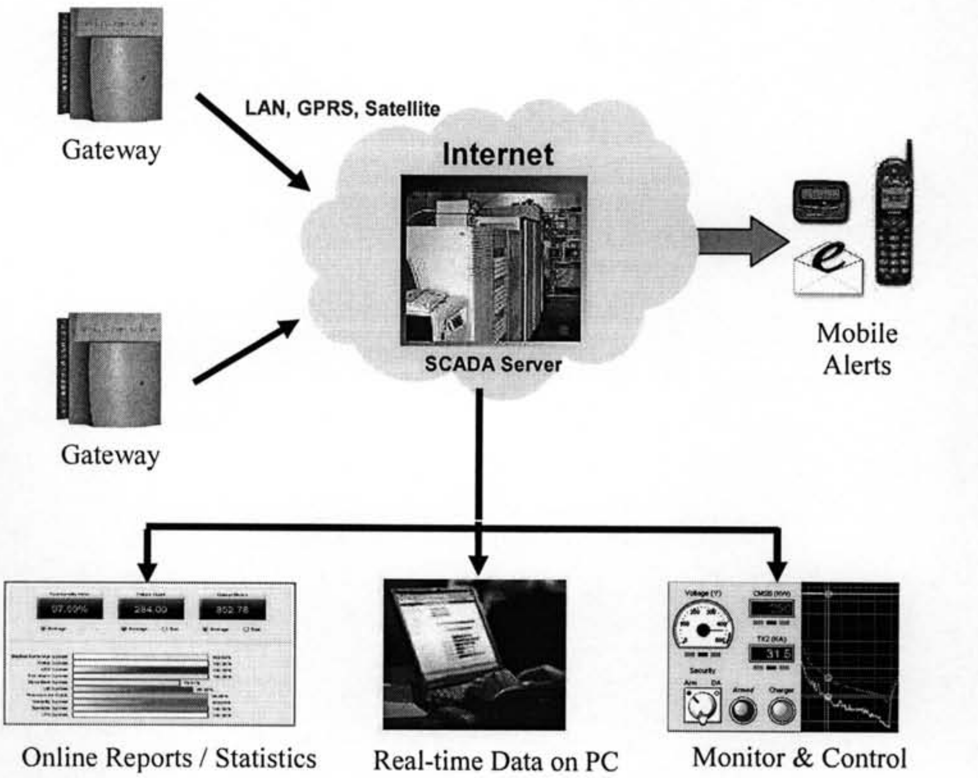


Figure 5.8 Suggested System Architecture for Further Study

The communication network from the data points might not have to be limited to Ethernet network only, but it can be GPRS or even Satellite. This, together with the improved interface as suggested above, we can have the advanced system architecture as shown in Figure 5.8. With this architecture, we can achieve the full features of an Internet-based solution for SCADA systems with a more professional and more interactive interface.

- **For the DS module**, in this research, it directly calls the Matlab program each time it receives a command. Matlab works independently from the DS program, it is only called upon the arrival of a new command. Therefore, the Matlab program is required to be installed in the computer where the DS module is installed. However, this method gives a fast and accurate response of the DS module to system users.

Future study on this simulation scheme can still use Matlab. However, Matlab 6.5 and above are equipped with the COM compiler. Any m-file written under .m format can be translated into a COM object which is compatible in both .NET and Java environment. Therefore, the DS module can call the COM object for the calculating purpose upon receiving a command. This tool is called *comtool*; it can be called directly from the Matlab command window to compile the DS's m-files into COM objects. This tool has been tested, but the results did not show up fast and accurately. Therefore, more time exploring this idea should be spent.