CHAPTER IV

SYSTEM IMPLEMENTATION

Continued from Chapter 3, this chapter will present all the details about the system implementation. All the features proposed in Chapter 3 are analyzed from the implementation point of view. Details about all system features stated and the implementation steps to achieve these features will be discussed in this chapter. Specific hardware and software tools will be taken into each functional description in order to make every feature statement clear. A conclusion for this chapter and also for all the presented chapters is given in Chapter 5.

This chapter will go first with the tentative implementation architecture following by a general analysis. The second part of this chapter, which is the most valuable and important part, is the experimental system with all information about the hardware devices and equipments as well as the corresponding software tools to be used for each system's function. Next, section 4.3 in this chapter is the presentation of the test results in each system's feature. All the figures presenting the actual operation of the system can be found here.

4.1 Tentative Architecture

Figure 4.1 below is intended for building a real system with hardware devices and software tools. This scheme is almost the same for most SCADA systems. SCADA systems are different in the deployed communication network and in the specific applications they are used for. For example, in substation automation, optical fiber is used to provide the immunity to interference. This ensures the accuracy of system's data and the SCADA center is often far away from the field devices.

In this research, Internet is used as the communication network and the SCADA center lies within the Ethernet LAN network with field devices. The solution proposed in this research is most suitable for such applications as Fuel Unloading/storage and Feed, Water Supply Monitoring, Irrigation, Flood Warning, Oil and Gas Tracking. The public nature of Internet does not allow us to maintain the data transfer speed consistent at different points of time in a day. Therefore, the proposed solution might not be suitable to be applied on such applications which require a strict real-time criterion as Power System Failure Detection system or other kinds of emergency reaction systems.

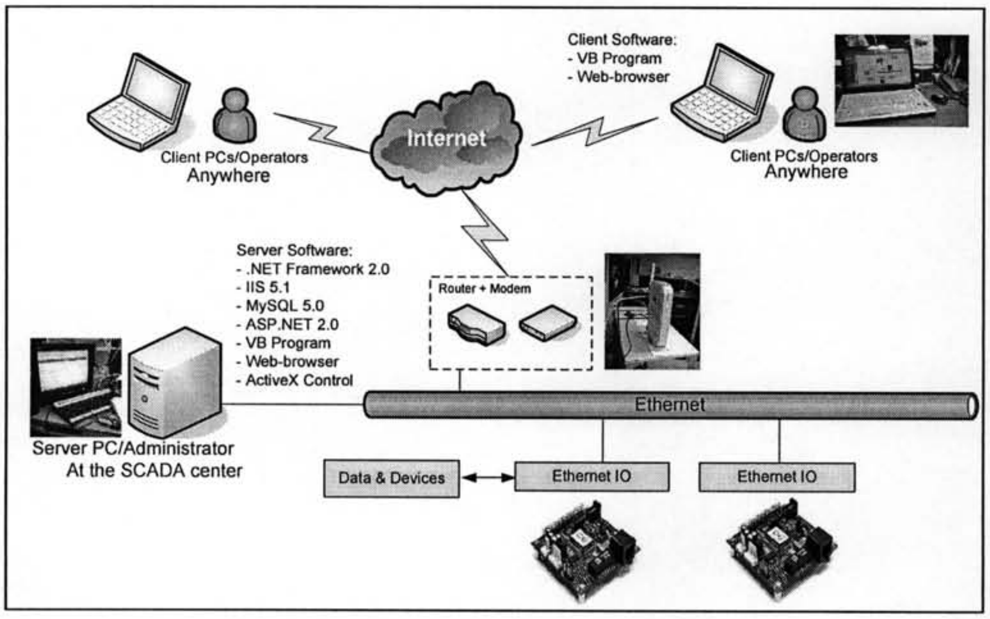


Figure 4.1 Tentative Implementation Architecture

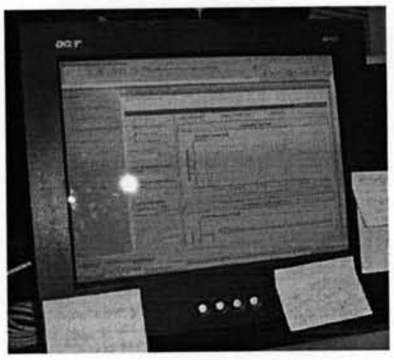
1.10 Implementation Details

In this thesis, for the illustration purpose, the simulated devices are going to be used. As mentioned in Chapter 3, more security tools will be added for higher security level and better management. All parts of the system remain unchanged except the field devices. Here, in the thesis illustration, a software module called Device Simulator (DS) will be developed to functionally replace the role of the Ethernet IO board and its corresponding device. Section 4.2 will give details about both hardwares and software modules to be built in the system. DS is depicted in the last part of this section.

4.2.1 Hardwares

1. Server Computer

The server computer plays the most important role in the system. In the experimental system, a desktop computer with the following specifications is considered to be stable enough to serve as the system's server.



Specifications:

- Manufactured by ACER
- Operating system:
  Windows XP Service Pack 2
- CPU Speed: 3 GHz
- RAM Memory: 1 GHz

Figure 4.2 Server Computer

2. **Client Computer**

There can be more than one client user having actions on the system. Therefore, any computer which has been authenticated can act as the client computer. Only an Internet browser is required. In this demonstration, a laptop will be used as the client user for the testing purpose.

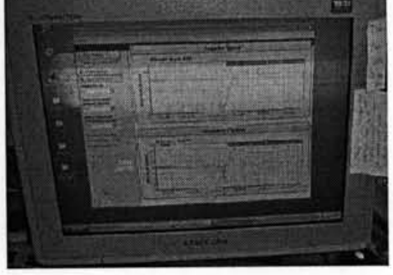
Specifications:

- Manufactured by SONY
- Operating system: Windows XP SP 2
- CPU Speed: 3 GHz
- RAM Memory: 1 GHz

Figure 4.3 Client Computer

3. **DS Computers**

This is the computer where the DS module is installed. The IP addresses of this computers will act as the IP addresses of the Ethernet IO boards.



Specifications:

- Manufactured by ASUS
- Operating system: Windows XP SP 2
- CPU Speed: 3 GHz
- RAM Memory: 1 GHz

Figure 4.4 DS Computer

4. Router + Modem

A **Netgear** router, model DG834G, was selected. This router contains a built-in modem. It has the task of connecting the system with one ISP (Internet Service Provider), therefore making the system available on the Internet.

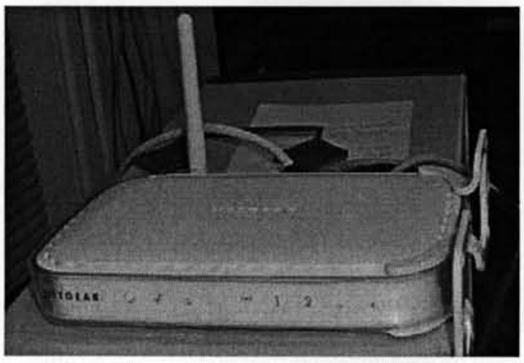


Figure 4.5 Router + Modem

Important features:

- Firewall

- The Ability to Enable or Disable IP Address Sharing by NAT (Network Address Translation)

- Automatic Configuration of Attached Computers by DHCP (Dynamic Host Configuration Protocol)

- Domain Name Server (DNS) Proxy & Dynamic DNS

- Classical IP (RFC 1577), PPP over Ethernet (PPPoE), PPP over ATM (PPPoA)

- Built-in ADSL modem

4.2.2 Software Tools and Modules

As discussed, at the device side, the Ethernet IO board is used to connect to the real devices to illustrate the stated functions of the proposed system. In the test system, the Ethernet IO board and the devices are going to be simulated using software. Therefore, we have three main parts in the system software including: Database (DB), SCADA Interface, and Device Simulator (DS).

The complete testing architecture of the system is shown as in Figure 4.6.

From Figure 4.6, we can see clearly what software tools we need to implement the application software and the database for the server computer. The two DS modules act as the Ethernet IO boards to which the devices have been connected.

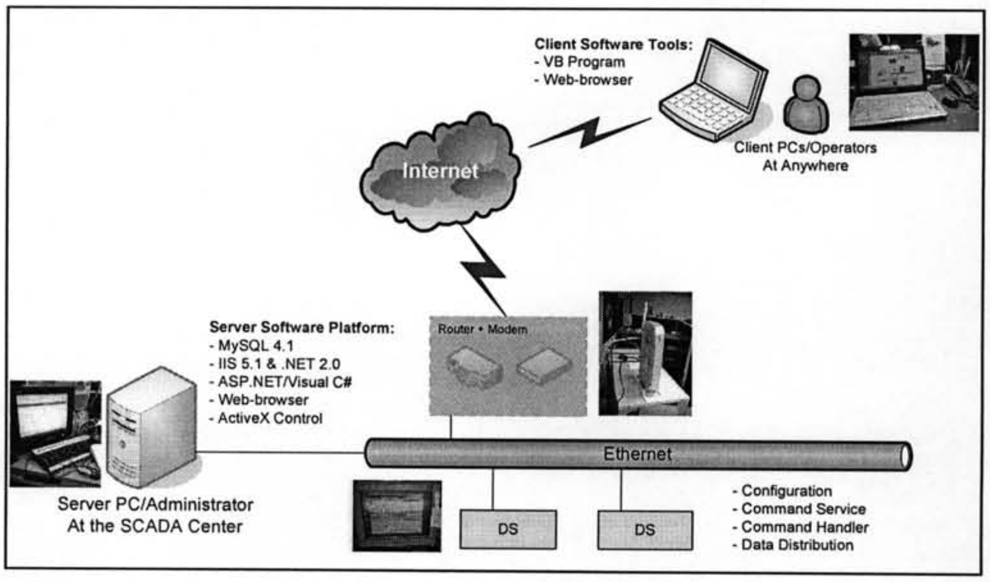


Figure 4.6 Testing System

There are several software platforms and tools needed to be used in building the system software which are Internet Information Services (IIS), .NET Framework, ASP.NET (ASP stands for Active Server Page) and MySQL. In this system, version 5.1 is used for IIS, 2.0 for .NET Framework; version 2.0 is the version for ASP.NET and MySQL is used with version 4.1. VC# is the programming language to be used.

Overall Communication Mechanism:

The proposed solution is built as a web application. It means that every access to the system is possible done via Internet. Here, approximately we classify two kinds of access to the system.

1. Access from server computer

From the system architecture (Figure 4.6), there is only one server computer. The server computer is connected to many devices in an Ethernet network. The server computer together with all the connected devices will then be placed at the SCADA control center, which is managed by the sytem administator. The Ethernet network allows a wide range of the number of entities and the connection distance which is enough for most industrial processes and plants.

In this solution, there is one computer which serves as both the database server and application server. Database and the application sofware which provides all functions on the system are located here. For an application which is not huge and complicated, it is a good idea to make use of one computer for both the database and the application software. By doing so, we can reduce the system's cumbersomeness, therefore reducing the implementation cost. The processing speed will also be significantly improved since the application software does not need to "travel a long way", in case the database server is a separate computer and it is located somewhere else on the Internet, to access the database.

For the illustration of the system proposed in this thesis, we have another computer where the DS module is installed. The communication between the server computer and the DS module which represents the device world is done via TCP/IP protocol over Ethernet network. The Ethernet network is very reliable in term of ensuring a stable communication link and it also provides the very fast speed, which helps make sure the real-time manner of all transactions in the system.

2. Access from client computers

The application software will be built as a website which is hosted in the server running 24 hours a day and is made available on the Internet. It provides the ability for the remote users to access to the system anytime and from any place with an Internet connection. Once a user has been logged on, the inteface will appear exactly the same as the interface on the server computer, the user then can operate the sytem as if he or she is sitting at the server computer. However, depending on his or her role on the system, a number of permissions will be given by the administrator. Again, all the transactions made by remote users are done via TCP/IP protocol. The communication range is unlimited to any Internet-connected place. Once a remote user has passed all the security restrictions, he or she can then access to the internal mechanism of the system and perform all the provided actions.

Communications are done over Internet network and we face the problem of Internet congestion and traffic. This will be made clear in the later part in this chapter.

A. Database

The system's database is built using MySQL 4.1. This part is going to present all about a database and all the data which are used in building the system's software.

- ***About MySQL 4.1***

- A database management system.

Generally, a database is a structured collection of data. It may be anything from a small amount of data of a single user to the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, we need a database management system such as MySQL Server. In helping computers handle large amounts of data, the database management system plays the central role in computing, as a standalone utility, or as a part of other applications.

- A relational database management system.

A relational database stores data in separate tables rather than putting all the data in one big storeroom. This adds speed and flexibility. The SQL part of "MySQL" stands for "Structured Query Language." SQL is the most common standardized language used to access databases and is defined by the ANSI/ISO SQL Standard.

- Open Source.

Open Source means that it is possible for anyone to both use and modify the software. We may also study the source code and can even change the MySQL system itself.

- MySQL Server works in client/server or embedded systems.

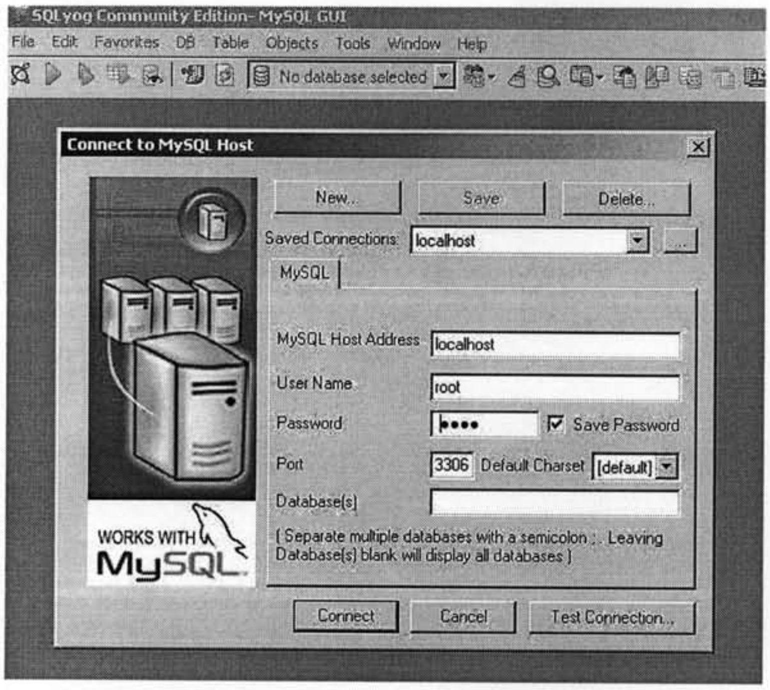
The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

o ***Working with MySQL 4.1***

With MySQL environment, every command is entered from the command line of SQL client. Several interface utilities has been developed for easy-to-use purpose of MySQL. SQLyog is an example. MySQL users only need to create a database framework, specify the connection parameters and connect to that database from SQLyog. Everything regarding creating tables, assigning indexes or database managing can be done via this graphical user interface.

To connect to a database in a computer which acts as the local host for the database is presented in Figure 4.7.

As shown in this figure, we have to specify some parameters. The field MySQL Host Address is the place where you store your system DB. Here, "localhost" means that we are storing all the system data in the DB in the local server computer. Username, Password and a Connection Port also have to be provided.



Saved Connections: localhost

MySQL Host Address: localhost

User Name: root

Password: •••• ☑ Save Password

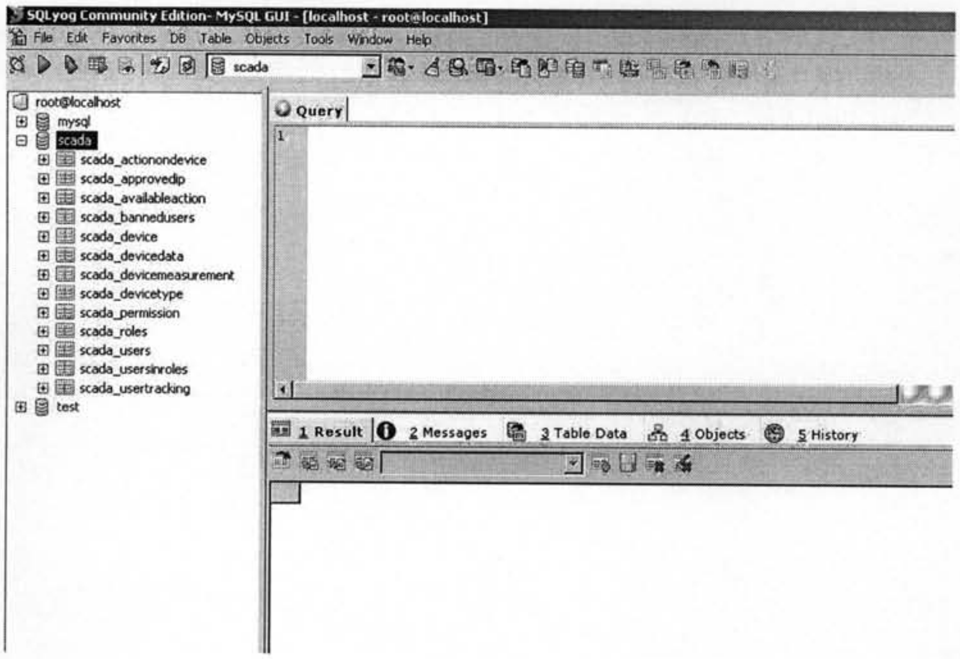
Port: 3306 Default Charset: [default]

Database(s):

(Separate multiple databases with a semicolon ; . Leaving Database(s) blank will display all databases)

Figure 4.7 Connecting to MySQL

After successfully connecting to a database, we have the following screen:


SQLyog Community Edition- MySQL GUI - [localhost - root@localhost]
File Edit Favorites DB Table Objects Tools Window Help
scada
Query
1 Result
2 Messages
3 Table Data
4 Objects
5 History

```
root@localhost
 mysql
 scada
   scada_actionondevice
   scada_approvedip
   scada_availableaction
   scada_bannedusers
   scada_device
   scada_devicedata
   scada_devicemeasurement
   scada_devicetype
   scada_permission
   scada_roles
   scada_users
   scada_usersinroles
   scada_usertracking
 test
```

Figure 4.8 MySQL Workspace

MySQL users need to insert all the source codes of their database into the Query window and execute the source codes. All the database tables will then be displayed on the left-hand column. The window below the Query window is the working space where allowing all the data viewing and customization.

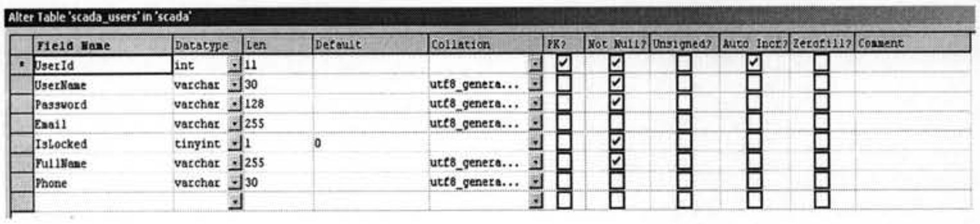
- ***Tables and Indexes to be used in the thesis***

Instead of storing all the system's data in a common storeplace or in a single file, data of system are divided into specific categories and stored in saparate tables. Each table represents one type of information and all the tables make up the system's data. Each table will then be accessed and handled by the application software on each specific task. In the thesis testing, we have the tables which appear in the left-hand column of Figure 4.8 above. Definition, function and MySQL source code of each table are given in the followings:

➢ **For User Management**

1. Scada_users

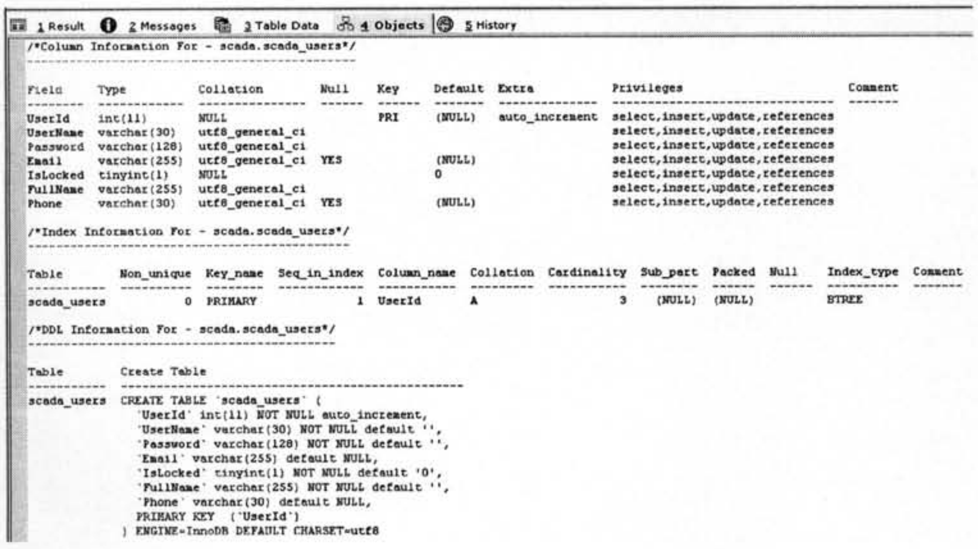
- This table defines the profile information relating to each user.
- Each user conforms to the following fields: UserId, UserNamse, Password, Email, IsLocked, FullName, and Phone.
- Technical specifications of each field:



Alter Table 'scada_users' in 'scada'

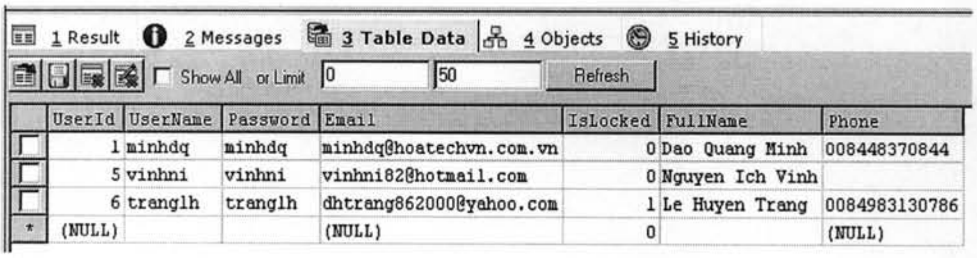
Field Name	Datatype	Len	Default	Collation	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
UserId	int	11			✓	✓	☐	✓	☐	
UserName	varchar	30		utf8_genera...	☐	✓	☐	☐	☐	
Password	varchar	128		utf8_genera...	☐	✓	☐	☐	☐	
Email	varchar	255		utf8_genera...	☐	☐	☐	☐	☐	
IsLocked	tinyint	1	0		☐	✓	☐	☐	☐	
FullName	varchar	255		utf8_genera...	☐	✓	☐	☐	☐	
Phone	varchar	30		utf8_genera...	☐	☐	☐	☐	☐	
					☐	☐	☐	☐	☐	

- Source code:



```
/*Column Information For - scada.scada_users*/
-----------------------------------------------

Field     Type          Collation        Null    Key     Default  Extra           Privileges                       Comment
--------  ------------  ---------------  ------  ------  -------  --------------  -------------------------------  -------
UserId    int(11)       NULL                     PRI     (NULL)   auto_increment  select,insert,update,references
UserName  varchar(30)   utf8_general_ci                                           select,insert,update,references
Password  varchar(128)  utf8_general_ci                                           select,insert,update,references
Email     varchar(255)  utf8_general_ci  YES             (NULL)                   select,insert,update,references
IsLocked  tinyint(1)    NULL                             0                        select,insert,update,references
FullName  varchar(255)  utf8_general_ci                                           select,insert,update,references
Phone     varchar(30)   utf8_general_ci  YES             (NULL)                   select,insert,update,references

/*Index Information For - scada.scada_users*/
----------------------------------------------

Table        Non_unique  Key_name  Seq_in_index  Column_name  Collation  Cardinality  Sub_part  Packed  Null    Index_type  Comment
-----------  ----------  --------  ------------  -----------  ---------  -----------  --------  ------  ------  ----------  -------
scada_users           0  PRIMARY              1  UserId       A                    3    (NULL)  (NULL)          BTREE

/*DDL Information For - scada.scada_users*/
--------------------------------------------

Table        Create Table
-----------  ----------------------------------------------------
scada_users  CREATE TABLE `scada_users` (
               `UserId` int(11) NOT NULL auto_increment,
               `UserName` varchar(30) NOT NULL default '',
               `Password` varchar(128) NOT NULL default '',
               `Email` varchar(255) default NULL,
               `IsLocked` tinyint(1) NOT NULL default '0',
               `FullName` varchar(255) NOT NULL default '',
               `Phone` varchar(30) default NULL,
               PRIMARY KEY  (`UserId`)
             ) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

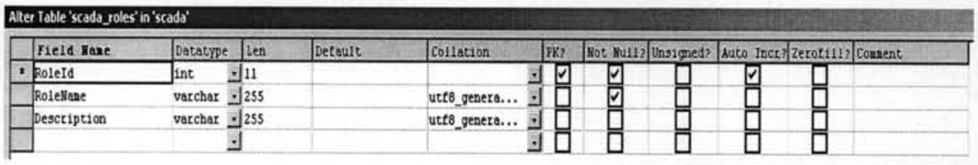
- Actual data stored which will then be used in the system:



	UserId	UserName	Password	Email	IsLocked	FullName	Phone
☐	1	minhdq	minhdq	minhdq@hoatechvn.com.vn	0	Dao Quang Minh	008448370844
☐	5	vinhni	vinhni	vinhni82@hotmail.com	0	Nguyen Ich Vinh	
☐	6	tranglh	tranglh	dhtrang862000@yahoo.com	1	Le Huyen Trang	0084983130786
*	(NULL)			(NULL)	0		(NULL)

2. *Scada_roles*

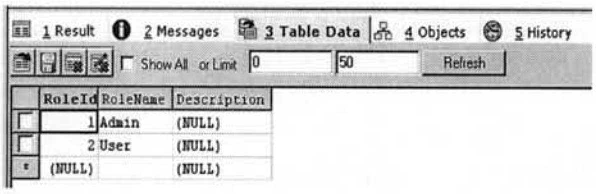
- This table defines the role of each user in the system.
- Each role conforms to two fields which are RoleId and RoleName.



Alter Table 'scada_roles' in 'scada'

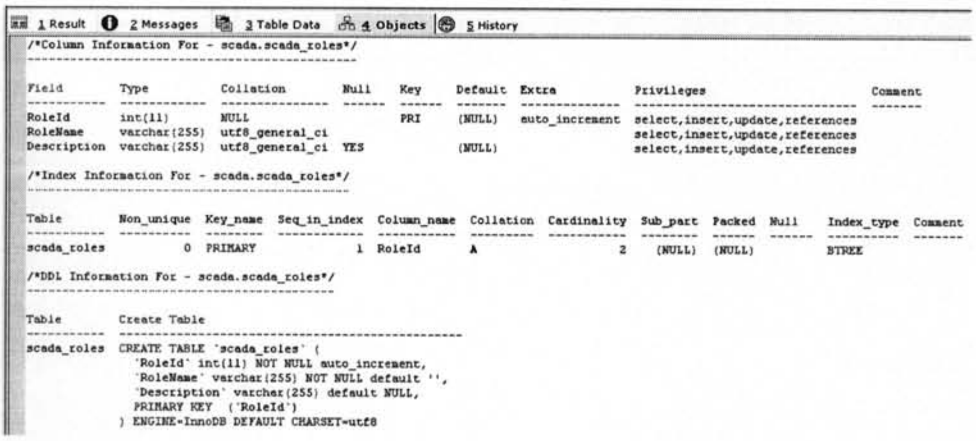
Field Name	Datatype	Len	Default	Collation	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
RoleId	int	11			✓	✓	☐	✓	☐	
RoleName	varchar	255		utf8_genera...	☐	✓	☐	☐	☐	
Description	varchar	255		utf8_genera...	☐	☐	☐	☐	☐	
					☐	☐	☐	☐	☐	

- The system defines two roles for users which are Administrator (Admin, for short) and normal users (user).



	RoleId	RoleName	Description
☐	1	Admin	(NULL)
☐	2	User	(NULL)
*	(NULL)		(NULL)

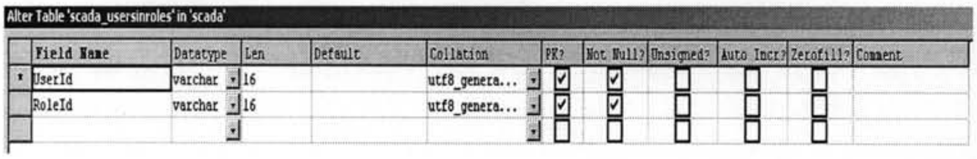
- Source Code:



```
/*Column Information For - scada.scada_roles*/
------------------------------------------------

Field        Type          Collation        Null    Key     Default  Extra           Privileges                        Comment
-----------  ------------  ---------------  ------  ------  -------  --------------  --------------------------------  -------
RoleId       int(11)       NULL                     PRI     (NULL)   auto_increment  select,insert,update,references
RoleName     varchar(255)  utf8_general_ci                                           select,insert,update,references
Description  varchar(255)  utf8_general_ci  YES             (NULL)                   select,insert,update,references

/*Index Information For - scada.scada_roles*/
-----------------------------------------------

Table        Non_unique  Key_name  Seq_in_index  Column_name  Collation  Cardinality  Sub_part  Packed  Null    Index_type  Comment
-----------  ----------  --------  ------------  -----------  ---------  -----------  --------  ------  ------  ----------  -------
scada_roles           0  PRIMARY              1  RoleId       A                    2    (NULL)  (NULL)          BTREE

/*DDL Information For - scada.scada_roles*/
---------------------------------------------

Table        Create Table
-----------  ------------------------------------------------
scada_roles  CREATE TABLE `scada_roles` (
               `RoleId` int(11) NOT NULL auto_increment,
               `RoleName` varchar(255) NOT NULL default '',
               `Description` varchar(255) default NULL,
               PRIMARY KEY  (`RoleId`)
             ) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

3. *Scada_usersinroles*

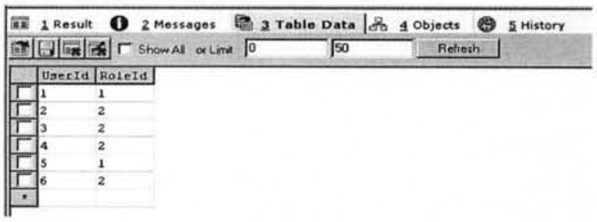
- This table defines which user belongs to which role in the system
- For each user, which has a User Id, he or she needs to have a Role Id, which can be either Admin or User. Therefore, we have UserId and RoleId in this table:



Alter Table 'scada_usersinroles' in 'scada'

Field Name	Datatype	Len	Default	Collation	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
UserId	varchar	16		utf8_genera...	✓	✓	☐	☐	☐	
RoleId	varchar	16		utf8_genera...	✓	✓	☐	☐	☐	
					☐	☐	☐	☐	☐	

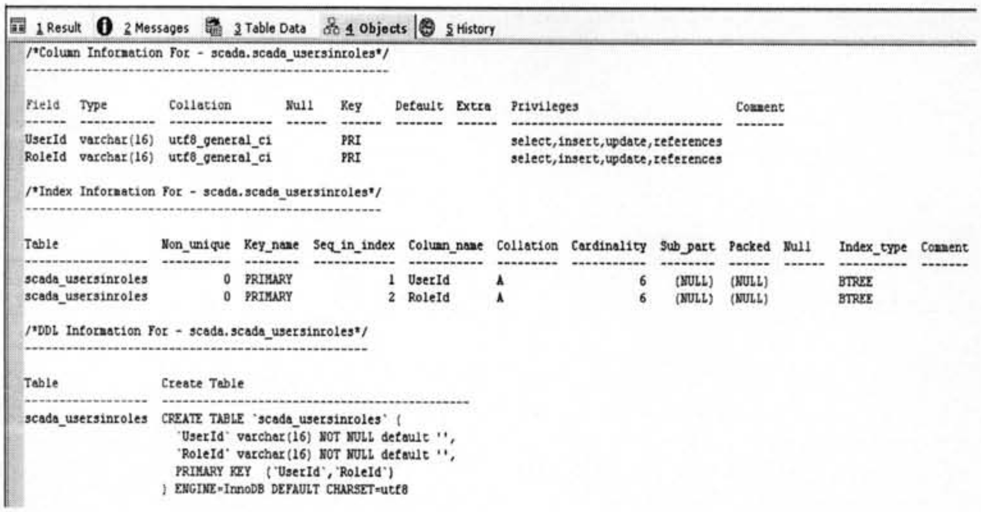
- Actual data:



UserId	RoleId
1	1
2	2
3	2
4	2
5	1
6	2

We see that the UserId represents the user which has been added to the system in an ascending order. The RoleId can be either 1 or 2 which represents the role of the user to be either Admin or User.

- Source Code:



```
/*Column Information For - scada.scada_usersinroles*/
----------------------------------------------------

Field   Type         Collation        Null    Key     Default  Extra   Privileges                       Comment
------  -----------  ---------------  ------  ------  -------  ------  -------------------------------  -------
UserId  varchar(16)  utf8_general_ci          PRI                      select,insert,update,references
RoleId  varchar(16)  utf8_general_ci          PRI                      select,insert,update,references

/*Index Information For - scada.scada_usersinroles*/
---------------------------------------------------

Table               Non_unique  Key_name  Seq_in_index  Column_name  Collation  Cardinality  Sub_part  Packed  Null    Index_type  Comment
------------------  ----------  --------  ------------  -----------  ---------  -----------  --------  ------  ------  ----------  -------
scada_usersinroles           0  PRIMARY              1  UserId       A                    6    (NULL)  (NULL)          BTREE
scada_usersinroles           0  PRIMARY              2  RoleId       A                    6    (NULL)  (NULL)          BTREE

/*DDL Information For - scada.scada_usersinroles*/
-------------------------------------------------

Table               Create Table
------------------  ---------------------------------------------
scada_usersinroles  CREATE TABLE `scada_usersinroles` (
                      `UserId` varchar(16) NOT NULL default '',
                      `RoleId` varchar(16) NOT NULL default '',
                      PRIMARY KEY  (`UserId`,`RoleId`)
                    ) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

➢ **For Data and Device Management**

1. Scada_device

- This table defines the devices in the system.
- Each device associates with the following fields of information:

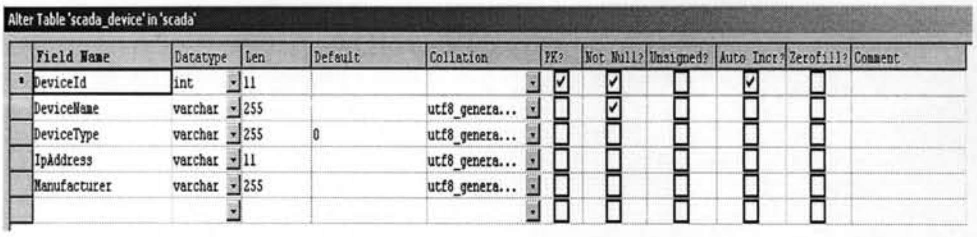
  . DeviceId: The ID of the device, defined by the system desiger.

  . DeviceType: The type of device, which might be Lamp or Motor, etc.

  . DeviceName: The name of device, defined by the system designer.

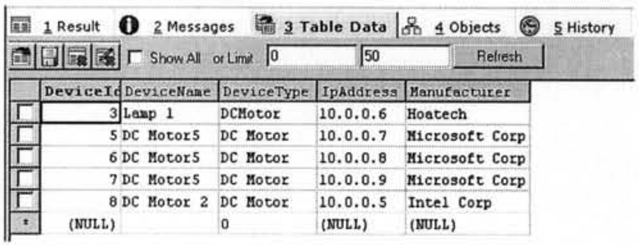
  . IpAddress: This is the IP address of the computer where DS is installed.

  . Manufacturer: Device's manufacturer, optional.
- All these fields are shown in the table below:


Alter Table 'scada_device' in 'scada'

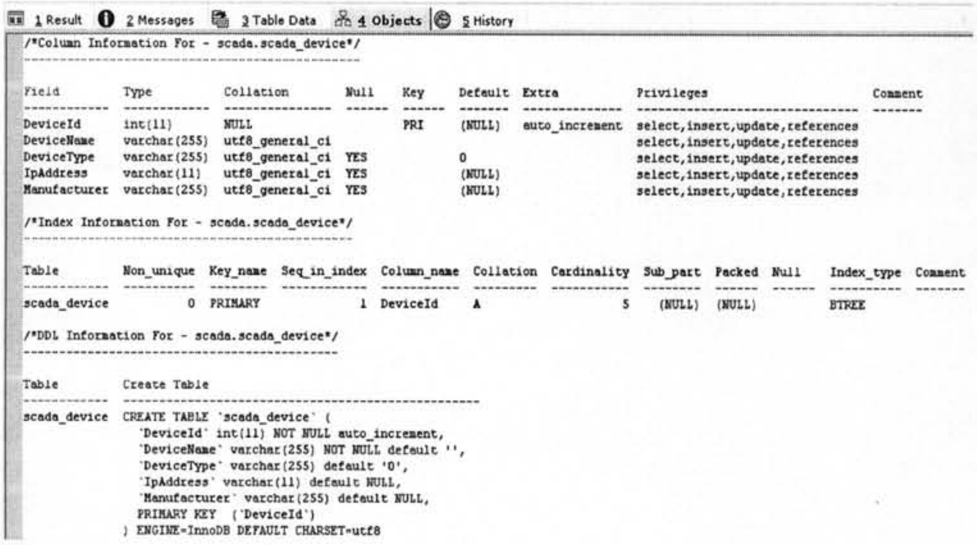
	Field Name	Datatype	Len	Default	Collation	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
*	DeviceId	int	11			✓	✓	☐	✓	☐	
	DeviceName	varchar	255		utf8_genera...	☐	✓	☐	☐	☐	
	DeviceType	varchar	255	0	utf8_genera...	☐	☐	☐	☐	☐	
	IpAddress	varchar	11		utf8_genera...	☐	☐	☐	☐	☐	
	Manufacturer	varchar	255		utf8_genera...	☐	☐	☐	☐	☐	
						☐	☐	☐	☐	☐	

- Actual data in the table, which will then be shown on the SCADA interface:



	DeviceId	DeviceName	DeviceType	IpAddress	Manufacturer
☐	3	Lamp 1	DCMotor	10.0.0.6	Hoatech
☐	5	DC Motor5	DC Motor	10.0.0.7	Microsoft Corp
☐	6	DC Motor5	DC Motor	10.0.0.8	Microsoft Corp
☐	7	DC Motor5	DC Motor	10.0.0.9	Microsoft Corp
☐	8	DC Motor 2	DC Motor	10.0.0.5	Intel Corp
*	(NULL)		0	(NULL)	(NULL)

- Source code:



```
/*Column Information For - scada.scada_device*/
------------------------------------------------

Field         Type          Collation        Null    Key     Default  Extra           Privileges                       Comment
------------  ------------  ---------------  ------  ------  -------  --------------  -------------------------------  -------
DeviceId      int(11)       NULL                     PRI     (NULL)   auto_increment  select,insert,update,references
DeviceName    varchar(255)  utf8_general_ci                                           select,insert,update,references
DeviceType    varchar(255)  utf8_general_ci  YES             0                        select,insert,update,references
IpAddress     varchar(11)   utf8_general_ci  YES             (NULL)                   select,insert,update,references
Manufacturer  varchar(255)  utf8_general_ci  YES             (NULL)                   select,insert,update,references

/*Index Information For - scada.scada_device*/
-----------------------------------------------

Table         Non_unique  Key_name  Seq_in_index  Column_name  Collation  Cardinality  Sub_part  Packed  Null    Index_type  Comment
------------  ----------  --------  ------------  -----------  ---------  -----------  --------  ------  ------  ----------  -------
scada_device           0  PRIMARY              1  DeviceId     A                    5    (NULL)  (NULL)          BTREE

/*DDL Information For - scada.scada_device*/
---------------------------------------------

Table         Create Table
------------  ---------------------------------------------------
scada_device  CREATE TABLE `scada_device` (
                `DeviceId` int(11) NOT NULL auto_increment,
                `DeviceName` varchar(255) NOT NULL default '',
                `DeviceType` varchar(255) default '0',
                `IpAddress` varchar(11) default NULL,
                `Manufacturer` varchar(255) default NULL,
               PRIMARY KEY  (`DeviceId`)
              ) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

2. *Scada_devicedata*

- This table contains the data of a specific device
- To make it specific to the data of one device, the following fields are needed:

  . Id: Defines each value of data added to the DB. Id ascendingly increases.

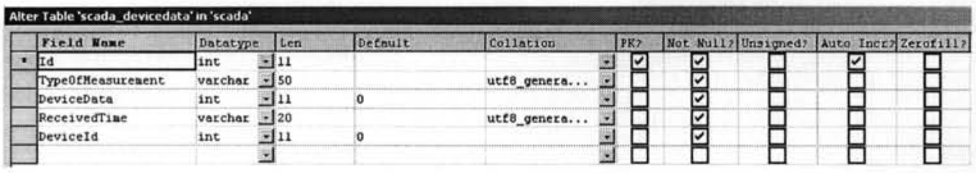
  . TypeOfMeasurement: This is the specific data to be monitored of each device. With a DC motor, this field can be either Angular Speed or Armature Current.

  . DeviceData: Each value of data coming from the device

  . ReceivedTime: The time at which the data is inserted into the DB

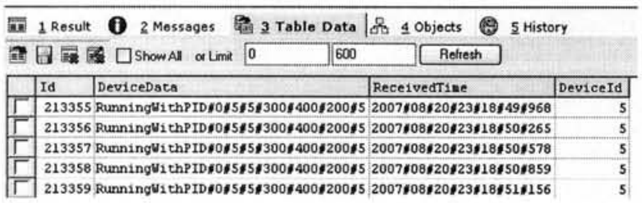
  . DeviceId: Is used to link to a specific device in the Scada_device table.

- All these fields are shown in the table below:


Alter Table 'scada_devicedata' in 'scada'

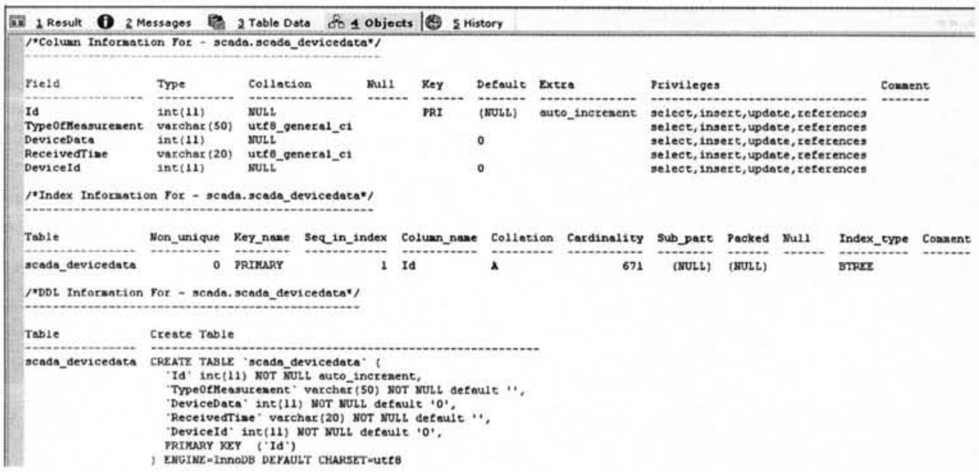
	Field Name	Datatype	Len	Default	Collation	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?
*	Id	int	11			✓	✓	☐	✓	☐
	TypeOfMeasurement	varchar	50		utf8_genera...	☐	✓	☐	☐	☐
	DeviceData	int	11	0		☐	✓	☐	☐	☐
	ReceivedTime	varchar	20		utf8_genera...	☐	✓	☐	☐	☐
	DeviceId	int	11	0		☐	✓	☐	☐	☐
						☐	☐	☐	☐	☐

- Actual data in this table might be as below:



	Id	DeviceData	ReceivedTime	DeviceId
☐	213355	RunningWithPID#0#5#5#300#400#200#5	2007#08#20#23#18#49#968	5
☐	213356	RunningWithPID#0#5#5#300#400#200#5	2007#08#20#23#18#50#265	5
☐	213357	RunningWithPID#0#5#5#300#400#200#5	2007#08#20#23#18#50#578	5
☐	213358	RunningWithPID#0#5#5#300#400#200#5	2007#08#20#23#18#50#859	5
☐	213359	RunningWithPID#0#5#5#300#400#200#5	2007#08#20#23#18#51#156	5

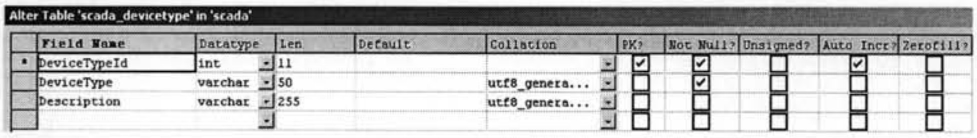
- Source code:



```
/*Column Information For - scada.scada_devicedata*/
--------------------------------------------------

Field              Type         Collation        Null    Key     Default  Extra           Privileges                        Comment
-----------------  -----------  ---------------  ------  ------  -------  --------------  --------------------------------  -------
Id                 int(11)      NULL                     PRI     (NULL)   auto_increment  select,insert,update,references
TypeOfMeasurement  varchar(50)  utf8_general_ci                                           select,insert,update,references
DeviceData         int(11)      NULL                             0                        select,insert,update,references
ReceivedTime       varchar(20)  utf8_general_ci                                           select,insert,update,references
DeviceId           int(11)      NULL                             0                        select,insert,update,references

/*Index Information For - scada.scada_devicedata*/
-------------------------------------------------

Table             Non_unique  Key_name  Seq_in_index  Column_name  Collation  Cardinality  Sub_part  Packed  Null    Index_type  Comment
----------------  ----------  --------  ------------  -----------  ---------  -----------  --------  ------  ------  ----------  -------
scada_devicedata           0  PRIMARY              1  Id           A                  671    (NULL)  (NULL)          BTREE

/*DDL Information For - scada.scada_devicedata*/
-----------------------------------------------

Table             Create Table
----------------  ------------------------------------------------------
scada_devicedata  CREATE TABLE `scada_devicedata` (
                    `Id` int(11) NOT NULL auto_increment,
                    `TypeOfMeasurement` varchar(50) NOT NULL default '',
                    `DeviceData` int(11) NOT NULL default '0',
                    `ReceivedTime` varchar(20) NOT NULL default '',
                    `DeviceId` int(11) NOT NULL default '0',
                    PRIMARY KEY  (`Id`)
                  ) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

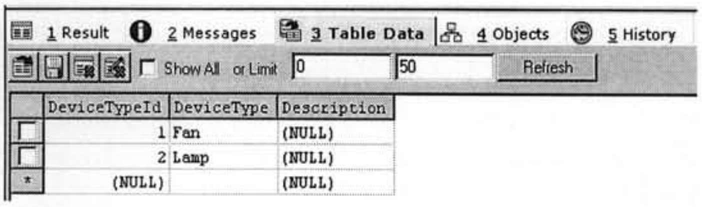
3. *Scada_devicetype*

- This table defines the types of the system's devices with three fields:
  . DeviceTypeld: Represents the device type, added in an ascending order
  . DeviceType: The specific type of each device
  . Description: The description of the type of the device
- All these fields are shown in the table below:


Alter Table 'scada_devicetype' in 'scada'

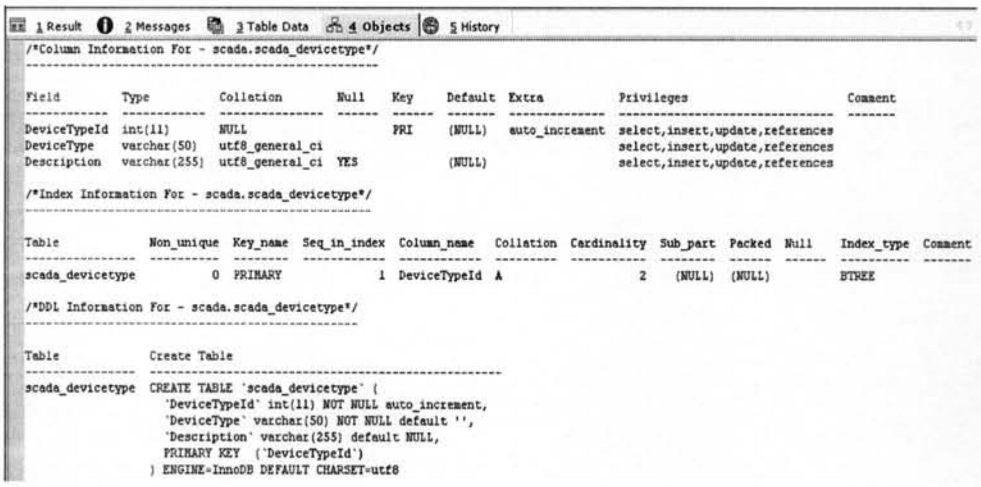
	Field Name	Datatype	Len	Default	Collation	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?
*	DeviceTypeId	int	11			✓	✓	☐	✓	☐
	DeviceType	varchar	50		utf8_genera...	☐	✓	☐	☐	☐
	Description	varchar	255		utf8_genera...	☐	☐	☐	☐	☐
						☐	☐	☐	☐	☐

- Actual data:



	DeviceTypeId	DeviceType	Description
☐	1	Fan	(NULL)
☐	2	Lamp	(NULL)
*	(NULL)		(NULL)

- Source code:



```
/*Column Information For - scada.scada_devicetype*/
---------------------------------------------------

Field         Type          Collation        Null    Key     Default  Extra           Privileges                        Comment
------------  ------------  ---------------  ------  ------  -------  --------------  --------------------------------  -------
DeviceTypeId  int(11)       NULL                     PRI     (NULL)   auto_increment  select,insert,update,references
DeviceType    varchar(50)   utf8_general_ci                                           select,insert,update,references
Description   varchar(255)  utf8_general_ci  YES             (NULL)                   select,insert,update,references

/*Index Information For - scada.scada_devicetype*/
--------------------------------------------------

Table             Non_unique  Key_name  Seq_in_index  Column_name   Collation  Cardinality  Sub_part  Packed  Null    Index_type  Comment
----------------  ----------  --------  ------------  ------------  ---------  -----------  --------  ------  ------  ----------  -------
scada_devicetype           0  PRIMARY              1  DeviceTypeId  A                    2    (NULL)  (NULL)          BTREE

/*DDL Information For - scada.scada_devicetype*/
------------------------------------------------

Table             Create Table
----------------  ---------------------------------------------------
scada_devicetype  CREATE TABLE `scada_devicetype` (
                    `DeviceTypeId` int(11) NOT NULL auto_increment,
                    `DeviceType` varchar(50) NOT NULL default '',
                    `Description` varchar(255) default NULL,
                    PRIMARY KEY  (`DeviceTypeId`)
                  ) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

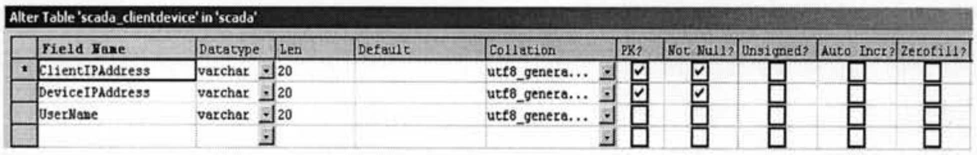
4. *Scada_clientdevice*

- This table stores the data of a device which is being operated by a client user.
- This table will then include three information fields which are:

  . Username: the username of the client user operating the system

  . ClientIPAddress: the IP address of the client computer operating the system

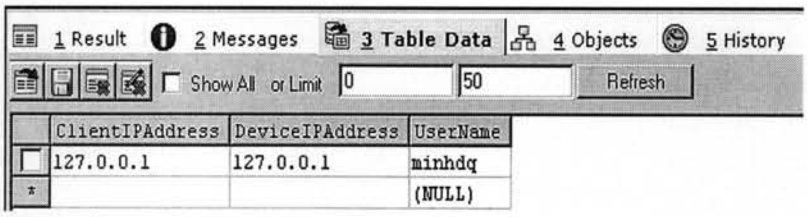
  . DeviceIPAddress: the IP address of the device being in used by the client.
- All these fields are shown in the table below:



Alter Table 'scada_clientdevice' in 'scada'

Field Name	Datatype	Len	Default	Collation	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?
ClientIPAddress	varchar	20		utf8_genera...	☑	☑	☐	☐	☐
DeviceIPAddress	varchar	20		utf8_genera...	☑	☑	☐	☐	☐
UserName	varchar	20		utf8_genera...	☐	☐	☐	☐	☐
					☐	☐	☐	☐	☐

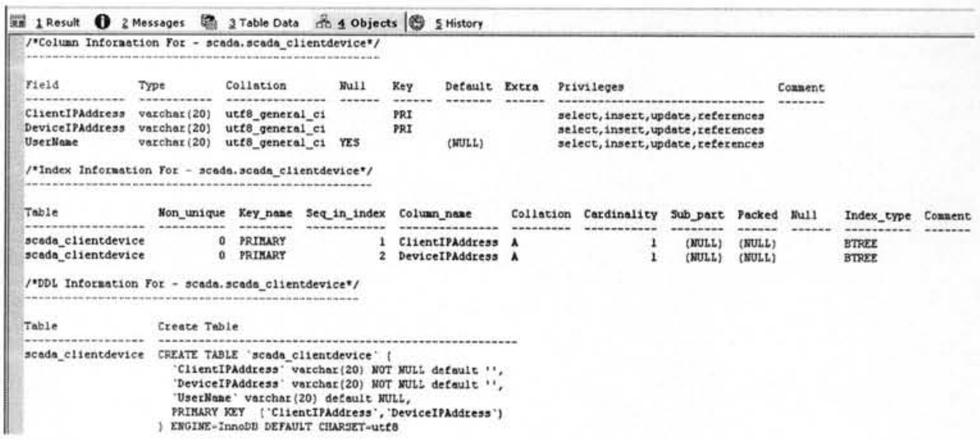
- Actual data in this table might be as below:



ClientIPAddress	DeviceIPAddress	UserName
127.0.0.1	127.0.0.1	minhdq
		(NULL)

From this table, we see that the client user currently operating the system has the username as *minhdq*. User *minhdq* is using the server computer to operate the system. Therefore, the ClientIPAddress and the DeviceIPAddress are the same with the value of 127.0.0.1. 127.0.0.1 is the default IP address of any server computer in a specific application.

- Source code:

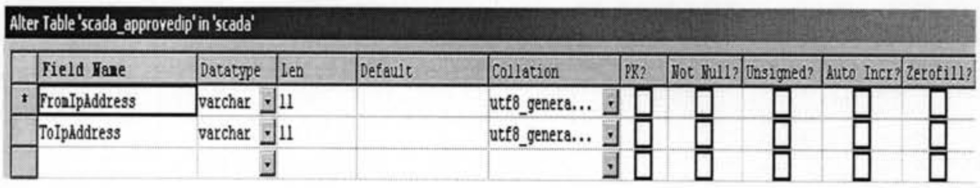


```
/*Column Information For - scada.scada_clientdevice*/
------------------------------------------------------

Field            Type          Collation        Null    Key     Default  Extra   Privileges                       Comment
---------------  ------------  ---------------  ------  ------  -------  ------  -------------------------------  -------
ClientIPAddress  varchar(20)   utf8_general_ci          PRI                      select,insert,update,references
DeviceIPAddress  varchar(20)   utf8_general_ci          PRI                      select,insert,update,references
UserName         varchar(20)   utf8_general_ci  YES              (NULL)          select,insert,update,references

/*Index Information For - scada.scada_clientdevice*/
-----------------------------------------------------

Table               Non_unique  Key_name  Seq_in_index  Column_name      Collation  Cardinality  Sub_part  Packed  Null    Index_type  Comment
------------------  ----------  --------  ------------  ---------------  ---------  -----------  --------  ------  ------  ----------  -------
scada_clientdevice           0  PRIMARY              1  ClientIPAddress  A                    1    (NULL)  (NULL)          BTREE
scada_clientdevice           0  PRIMARY              2  DeviceIPAddress  A                    1    (NULL)  (NULL)          BTREE

/*DDL Information For - scada.scada_clientdevice*/
---------------------------------------------------

Table               Create Table
------------------  -----------------------------------------------------
scada_clientdevice  CREATE TABLE `scada_clientdevice` (
                      `ClientIPAddress` varchar(20) NOT NULL default '',
                      `DeviceIPAddress` varchar(20) NOT NULL default '',
                      `UserName` varchar(20) default NULL,
                      PRIMARY KEY  (`ClientIPAddress`,`DeviceIPAddress`)
                    ) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

➢ **For Security Management**

1. Scada_approvedip

- This table defines the security tools by IP approval
- To manage user access by IP addresses, we need two information fields which are FromIpAddress and ToIpAddress as in the table below:



Alter Table 'scada_approvedip' in 'scada'

Field Name	Datatype	Len	Default	Collation	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?
FromIpAddress	varchar	11		utf8_genera...	☐	☐	☐	☐	☐
ToIpAddress	varchar	11		utf8_genera...	☐	☐	☐	☐	☐
					☐	☐	☐	☐	☐

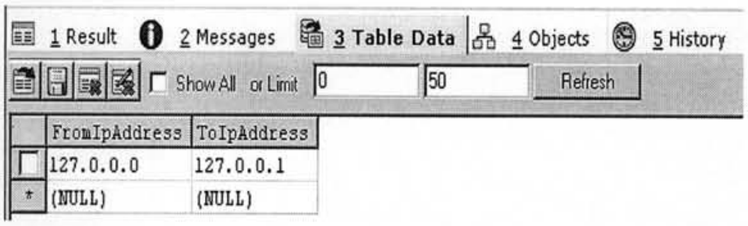
The admisnistrator can restrict user access by either one of these three options:

. The active machine. In this case, these two fields have the same value

. Limited IP range

. Everyone, meaning that every user can gain access to the system application software and these two fields have the NULL values.

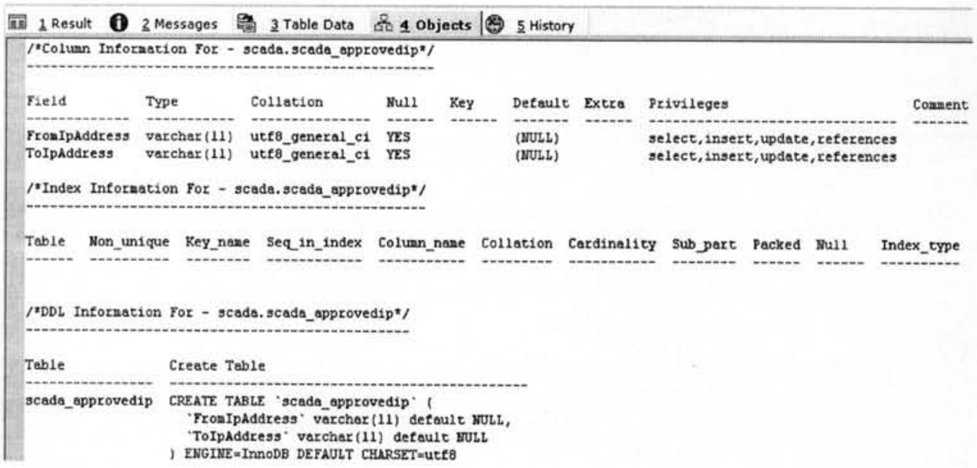
- Actual data:



FromIpAddress	ToIpAddress
127.0.0.0	127.0.0.1
(NULL)	(NULL)

We see the value 127.0.0.1 for both of two information fields in this table since the currently active computer is the server computer.

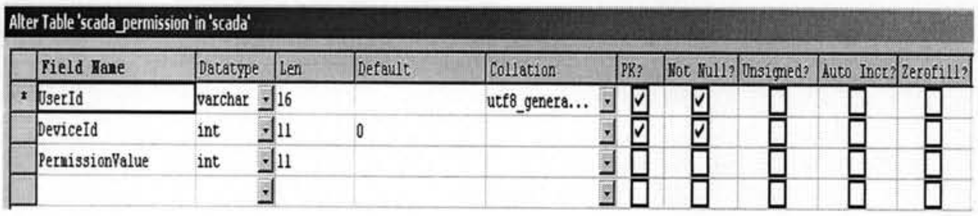
- Source code:



```
/*Column Information For - scada.scada_approvedip*/
--------------------------------------------------

Field         Type         Collation        Null   Key    Default Extra  Privileges                       Comment
------------  -----------  ---------------  ------ ------ ------- ------ -------------------------------  -------
FromIpAddress varchar(11)  utf8_general_ci  YES           (NULL)         select,insert,update,references
ToIpAddress   varchar(11)  utf8_general_ci  YES           (NULL)         select,insert,update,references

/*Index Information For - scada.scada_approvedip*/
-------------------------------------------------

Table  Non_unique  Key_name  Seq_in_index  Column_name  Collation  Cardinality  Sub_part  Packed  Null    Index_type
------ ----------  --------  ------------  -----------  ---------  -----------  --------  ------  ------  ----------


/*DDL Information For - scada.scada_approvedip*/
-----------------------------------------------

Table             Create Table
----------------  ---------------------------------------------
scada_approvedip  CREATE TABLE `scada_approvedip` (
                    `FromIpAddress` varchar(11) default NULL,
                    `ToIpAddress` varchar(11) default NULL
                  ) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

2. *Scada_permission*

- This table defines the user permission on each specific device
- Permissions mean the rights that a user is provided by the administrator. We need to know a user, a device and the action that the user wants to run on the device. Therefore, we have three information fields as below:



Alter Table 'scada_permission' in 'scada'

Field Name	Datatype	Len	Default	Collation	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?
UserId	varchar	16		utf8_genera...	✓	✓			
DeviceId	int	11	0		✓	✓			
PermissionValue	int	11							

The UserId is used to define the user, DeviceId is used to define the device and the PermissionValue is used to point out which action is to be granted for the user.

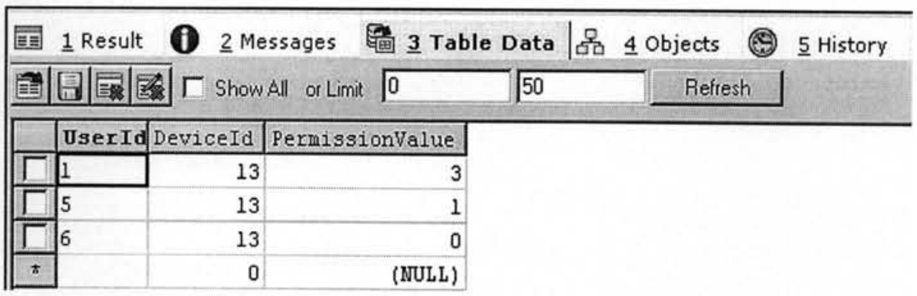
- PermissionValue is either one of the followings, depending on each user:

. 0: user has no right on the corresponding device

. 1: user has the right to view data of the corresponding device

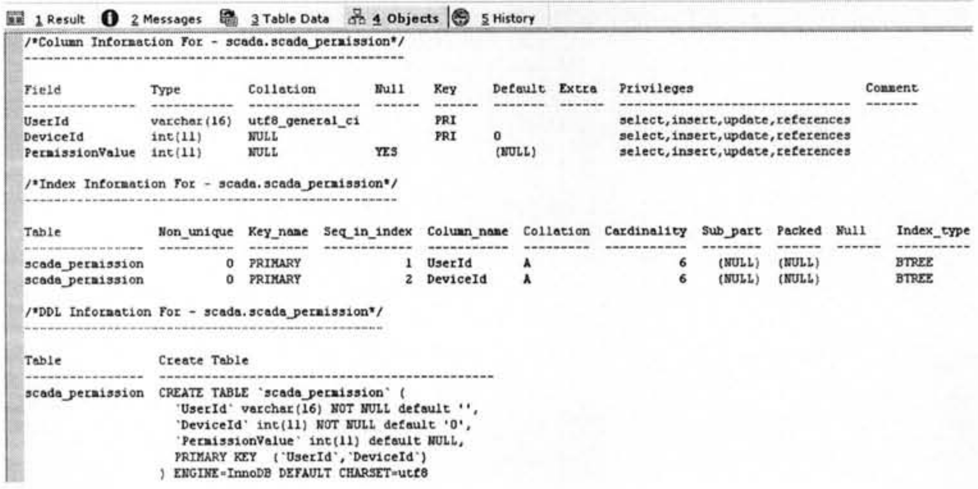
. 3: user can both view data of and manage the corresponding device

- Actual data:



UserId	DeviceId	PermissionValue
1	13	3
5	13	1
6	13	0
	0	(NULL)

- Source code:

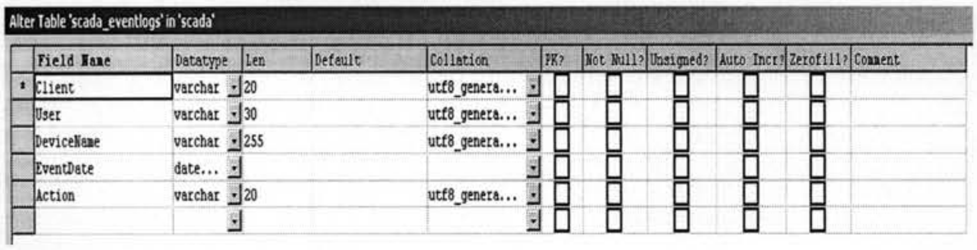


```
/*Column Information For - scada.scada_permission*/
--------------------------------------------------

Field           Type         Collation        Null    Key     Default  Extra   Privileges                       Comment
--------------- ------------ ---------------- ------- ------- -------- ------- -------------------------------- -------
UserId          varchar(16)  utf8_general_ci          PRI                      select,insert,update,references
DeviceId        int(11)      NULL                     PRI     0                select,insert,update,references
PermissionValue int(11)      NULL             YES             (NULL)           select,insert,update,references

/*Index Information For - scada.scada_permission*/
--------------------------------------------------

Table            Non_unique Key_name Seq_in_index Column_name Collation Cardinality Sub_part Packed Null   Index_type
---------------- ---------- -------- ------------ ----------- --------- ----------- -------- ------ ------ ----------
scada_permission          0 PRIMARY             1 UserId      A                   6   (NULL) (NULL)        BTREE
scada_permission          0 PRIMARY             2 DeviceId    A                   6   (NULL) (NULL)        BTREE

/*DDL Information For - scada.scada_permission*/
------------------------------------------------

Table            Create Table
---------------- ------------------------------------------
scada_permission CREATE TABLE `scada_permission` (
                   `UserId` varchar(16) NOT NULL default '',
                   `DeviceId` int(11) NOT NULL default '0',
                   `PermissionValue` int(11) default NULL,
                   PRIMARY KEY  (`UserId`,`DeviceId`)
                 ) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

➢ **Event Logs**

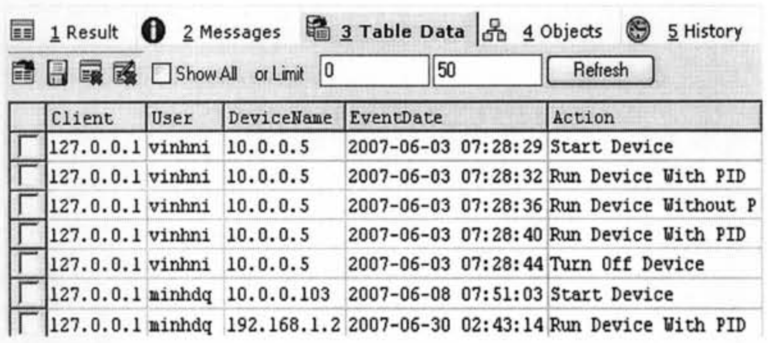
1. Scada_eventlogs

- This table defines all the actions happened and happening in the system. Once an action has been performed, it will be recorded in this table.
- In case of failure, this table's data will be of importance since it helps the system operator know which action might possibly the cause of the failure
- This table contains the following data fields:



Field Name	Datatype	Len	Default	Collation	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
Client	varchar	20		utf8_genera...	☐	☐	☐	☐	☐	
User	varchar	30		utf8_genera...	☐	☐	☐	☐	☐	
DeviceName	varchar	255		utf8_genera...	☐	☐	☐	☐	☐	
EventDate	date...				☐	☐	☐	☐	☐	
Action	varchar	20		utf8_genera...	☐	☐	☐	☐	☐	
					☐	☐	☐	☐	☐	

- Some sample actions recorded as below:



Client	User	DeviceName	EventDate	Action
127.0.0.1	vinhni	10.0.0.5	2007-06-03 07:28:29	Start Device
127.0.0.1	vinhni	10.0.0.5	2007-06-03 07:28:32	Run Device With PID
127.0.0.1	vinhni	10.0.0.5	2007-06-03 07:28:36	Run Device Without P
127.0.0.1	vinhni	10.0.0.5	2007-06-03 07:28:40	Run Device With PID
127.0.0.1	vinhni	10.0.0.5	2007-06-03 07:28:44	Turn Off Device
127.0.0.1	minhdq	10.0.0.103	2007-06-08 07:51:03	Start Device
127.0.0.1	minhdq	192.168.1.2	2007-06-30 02:43:14	Run Device With PID

In the EventLog table, the following data will be recorded:

. Client: the IP address of the active client.

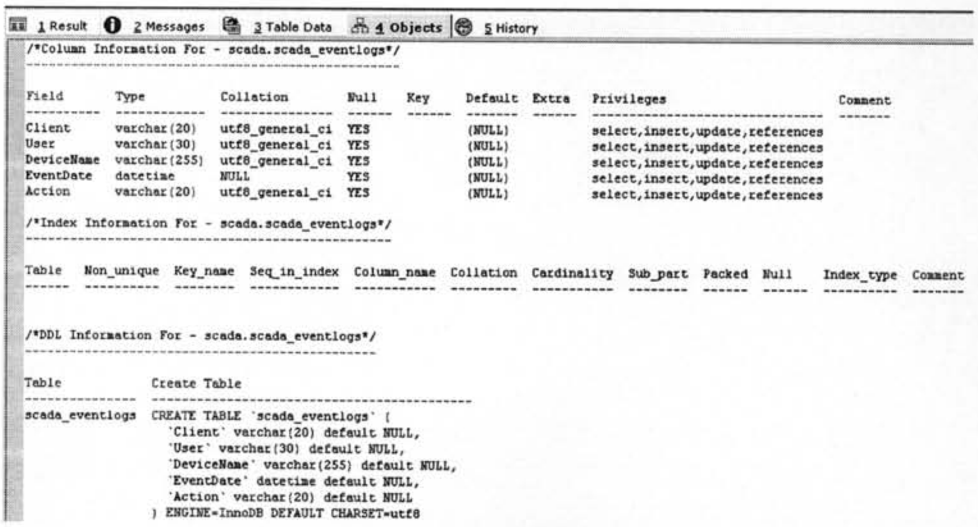
. User: the username corresponding to the active user

. DeviceName: the IP address of the computer where the DS module is installed.

. EventDate: date and time when the action takes place

. Action: the name of the recorded action

- Source code:



1 Result · 2 Messages · 3 Table Data · 4 Objects · 5 History

```
/*Column Information For - scada.scada_eventlogs*/
-------------------------------------------------

Field       Type          Collation        Null    Key     Default  Extra   Privileges                       Comment
----------  ------------  ---------------  ------  ------  -------  ------  -------------------------------  -------
Client      varchar(20)   utf8_general_ci  YES             (NULL)           select,insert,update,references
User        varchar(30)   utf8_general_ci  YES             (NULL)           select,insert,update,references
DeviceName  varchar(255)  utf8_general_ci  YES             (NULL)           select,insert,update,references
EventDate   datetime      NULL             YES             (NULL)           select,insert,update,references
Action      varchar(20)   utf8_general_ci  YES             (NULL)           select,insert,update,references

/*Index Information For - scada.scada_eventlogs*/
------------------------------------------------

Table   Non_unique  Key_name  Seq_in_index  Column_name  Collation  Cardinality  Sub_part  Packed  Null    Index_type  Comment
------  ----------  --------  ------------  -----------  ---------  -----------  --------  ------  ------  ----------  -------


/*DDL Information For - scada.scada_eventlogs*/
----------------------------------------------

Table            Create Table
---------------  -----------------------------------------
scada_eventlogs  CREATE TABLE `scada_eventlogs` (
                   `Client` varchar(20) default NULL,
                   `User` varchar(30) default NULL,
                   `DeviceName` varchar(255) default NULL,
                   `EventDate` datetime default NULL,
                   `Action` varchar(20) default NULL
                 ) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

All the contents in the system's DB have been presented in details. Nevertheless, whenever we design a DB system working in a real project, we have to keep in mind the capacticy of the DB. In almost all types of real-time systems, the system's server is supposed to be in operation 24 hours a day and 7 days a week. Therefore, the DB can be full during operation. As mentioned earlier, in a system, the application server and the DB server may either lie in two separated computers or in the same computer. The application software normally does not occupy much space, but the system's DB does. Depending on a specific application, the capacity of the DB should be carefully consired. It depends on the application requirements and the budget that the implementation unit can afford. It is apparent that the more space used for the DB, the more money we have to pay for both the hardware purchasing and maintenancing.

For such applications as Substation Automation, Fuel Pipe Tracking, Lighting, Building Automation, the DB might not necessarily be large since data in a long time operating are not required to be kept. In such applications, the system operation in one day may not have effect on the operation of the next day. Therefore, old data can be erased to release the space for the next data to be inserted in.

For such applications as Weather Forecast, Irrigation, Flood Warning, we have to have a big DB since much data are required to be stored during the operation period of up to 10 years, for the purpose of analyzing and giving future predictions. In these cases, the system designer must have the careful discussion with the implementation unit to define the operation period after which the data can be deleted out from the DB. Based on the specific application, the sytem designer may also give an estimation of the period of time for which the DB might be full. At that time, DB may be cleaned up to leave space for new data coming in.

In this research, one computer is used for both application server and DB server. 100MB is spent for the application software on C drive and about 40 GB for the DB on drive D. For the testing purpose, in this research, only two simulated DC motors plus the capacity used for other management functions are to be used. Thus, 40 GB is considered to be enough for 1 year experimenting and operating, from the start of the research.

B. Application software

This software part plays the most important role in the whole system. It provides not only the HMI between users and the system, but it also provides all the management as well as the functional features which ensure the smooth operation of the system.

1. User Management

There can be many users in the system. Therefore, a good management on users is a must. Generally, each user will be provided with one user account to be able to access the system. Each user must pass the username/password authentication step before giving any action on the system. This solution defines two types of users:

1. Administrator.

Administrator is the user who can fully control the system. The whole system has only one administrator at a time to avoid conflict which might happen if there are more than one user gaining the full control of the system and they have different policies on the same issue.

The administrator can create or delete accounts (users), track which devices are available, distribute or withdraw the right of other users. Besides, regarding the user management matter, the system's administrator can edit the information of existing users including Login name, Full name, Role, Email, Phone number.

Some users might have been locked from accessing the system while existing. The status of whether a user has been locked is also displayed on the User Management page. Administrator can lock or unlock an existing user through a link to the Security Mangement function. Permission on specific functions on the system might also be changed by the administrator. After successful login; the full name and also the role of the active user are displayed on the SCADA interface.

These jobs are done on web-interface, giving the administrator the ability to manage the system from any place in the world having the internet connection.

The designer will provide the first username and password to the system administrator. After the first successful login, the administrator may change that username and password for privacy reason.

2. *Normal users.*

Normal users only have the right to perform actions which have been defined or distributed by the administrator. By default, normal users are only able to view the system's data and devices' statuses, but not able to give any control command on the system. Certain rights of each normal user on the system will be approved by the administrator accordingly.

The User Management module is written using the model Object-Controller. For user management function, we have the user object and database tables for users. Each user corresponds to one record, in other words, one row, in the table of users. Source code (the controller) in VC# will be written to query data in the database and these data is to be filled into the object for further processing, meaning that the controller is mapping the object into the corresponding record. Vice versa, when a new user, meaning an object, is to be created, data for this user will be inserted into the DB for further use. This mechanism makes the software design, in general, more object-oriented, which is the most effective way to write application softwares.

Connections made to DB conform to ODBC* (Open Database Connectivity) standard. Therefore, the source code to control the object-database does not depend on what DB type (MySQL, SQL Server, Oracle, etc.) to be used, and the application software can support such these types of DB as well.

****ODBC**:*

ODBC is a standard software API specification for using database management systems (DBMS). ODBC is independent of programming language, database system and operating system. ODBC makes it possible to access any data from any application, regardless of which DBMS is handling the data. ODBC achieves this by inserting a middle layer called a database driver between an application and the DBMS. This layer translates the application's data queries into commands that the DBMS understands. ODBC standard has four components:

- ***Application***. Performs processing and calls ODBC functions to submit SQL statements and retrieve results.
- ***Driver Manager***. Loads and unloads drivers on behalf of an application. Processes ODBC function calls or passes them to a driver.
- ***Driver***. Processes ODBC function calls, submits SQL requests to a specific data source, and returns results to the application. If necessary, the driver modifies an application's request so that the request conforms to syntax supported by the associated DBMS.
- ***Data source***. Consists of the data the user wants to access and its associated operating system, DBMS, and network platform (if any) used to access the DBMS.

Figure 4.9 shows the relationship between these four components:

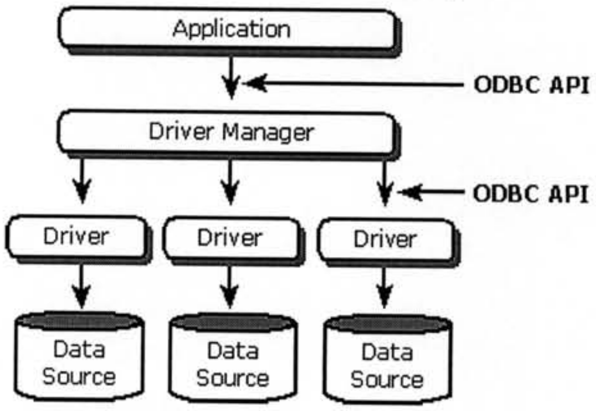


Figure 4.9 ODBC Architecture

Multiple drivers and data sources can exist, which allows the application to simultaneously access data from more than one data source.

The ODBC API is used in two places: between the application and the Driver Manager, and between the Driver Manager and each driver. The interface between the Driver Manager and the drivers is sometimes referred to as the service provider interface, or SPI. For ODBC, the application programming interface (API) and the service provider interface (SPI) are the same; that is, the Driver Manager and each driver have the same interface to the same functions.

The model Object-Controller and ODBC standard are not only used for managing the system's users, but for all other system's functions which need to access the DB. This way of handling system's data will go though all the system functionalities.

2. Device and Data Management

o **Data Management**

Data Management is one of the most important functions of any SCADA system. Generally speaking, system data contain all information about the entire system with the followings:

- Data generated by system's devices and equipments during operation. For example, device's output data or alarm, cautions, etc.
- Data inserted by system users/operators such as device's specifications, control commands, etc.

Here, ActiveX technology is used to collect data from the system device, store those data in the DB for further uses and display data on SCADA interface.

- *If the administrator is running the system at the server computer:*

This ActiveX is written using VB language and then it is embedded into asp (active server pages) in the server application software. This ActiveX also has the task of establishing connections to system devices (which are the DS modules mentioned). The overall principle of the operation of this ActiveX is illustrated in Figure 4.10:

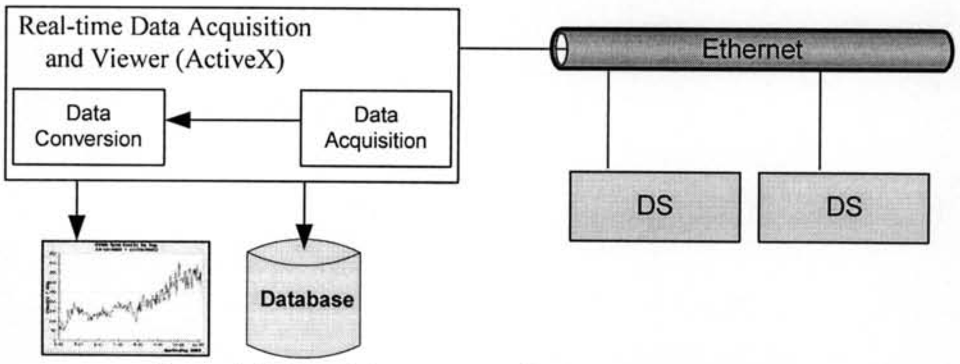


Figure 4.10 Data Management Principle from Server Side

Once the connection to a DS module has been established, this ActiveX will automatically collect data from the DS module and store these data in the system DB in a preset interval of 25 ms. In the mean time, these data are displayed in the form of continuously running graph and numerical format. Specifically, each DS module will have a separate set of data. Therefore, on the system interface, the name of the DS module which corresponds to the being-displayed graphs will also be displayed.

Communications between the server computer and the DS modules are done in Ethernet network.

- *If a normal user is running the system at a client computer:*

From the client side, the communication is going through the Internet network. It is established by http requests sent from the client computer to the server application. The overall principle is depicted in Figure 4.11.

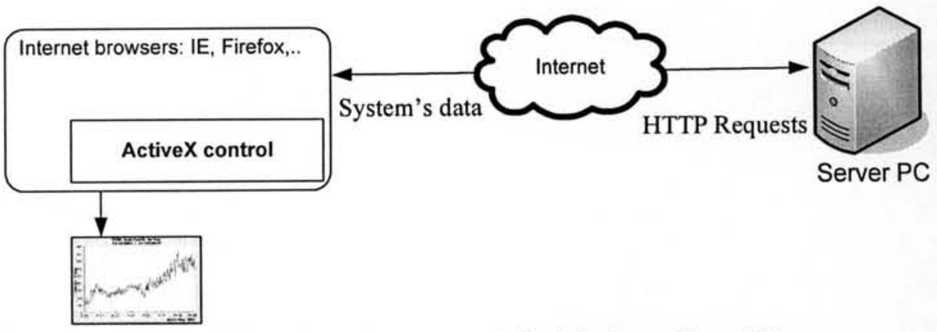


Figure 4.11 Data Management Principle from Client Side

From the web-browser at client computer, users need to connect to the server application software by entering the server's address into the address field of the Internet browser being used. Once the connection has been established successfully, the ActiveX control will be automatically loaded to the client's Internet browser. This allows the users at client side to run the system as if they are sitting at the server computer. The server, once it has received the requests from the AxtiveX, which is now residencing at the client computer, will forward those requests to the DS module as in the previous case. The server then receives the responses from the DS module and sends back the results to the client computer.

- **Data Transfer Speed and Time**

As mentioned in Chapter 1, the testing system in this research will show the speed and the time different from the devices to the client interface. This is to evaluate the performance of the testing system to consider which types of applications it fits.

Calculation Principle:

- t_1: the point of time of sending request from client interface
- t_2: the point of time when data are being displayed on client interface

$$\textbf{Variance Time} = t_2 - t_1$$

- TBX: total bytes of one-time exchanged between client and server (Sum of bytes of one request and one response)

$$\textbf{Data transfer Speed} = \frac{2*TBX}{t_2 - t_1}$$

- **History Data**

The thesis testing system provides users the ability to retrieve and view the historical data of system devices. Historical data are stored in the DB table scada_devicedata with specific date and time for each value (refer to section A. Database/For Data and Device Management). Once date and time has been chosen by users, these data will be taken from this table and displayed both graphically and numerically in tables. The length of the graphs is 60 seconds; users can choose the starting point from any time point in the past until present to have the display in each 60 second basis. All the data values falling in this 60 second period are displayed accordingly.

Again here, ActiveX technology is used to handle the system historical data.

- **Device Control**

The general principle for this function is presented in Figure 4.12.

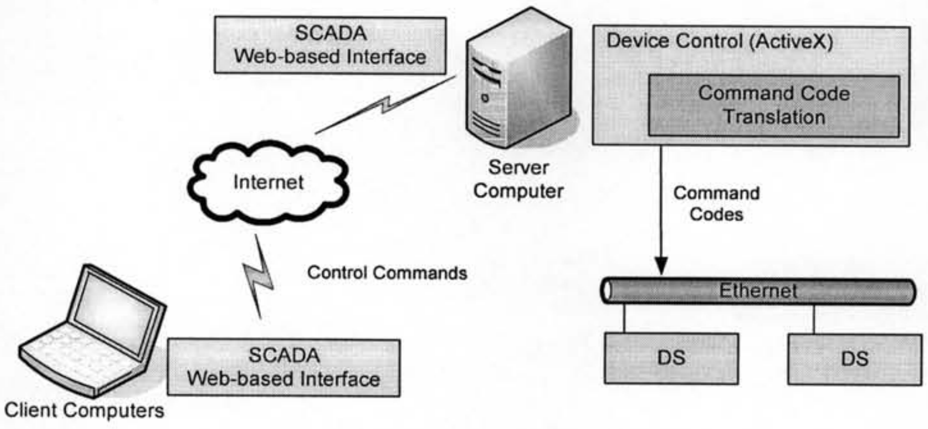


Figure 4.12 Device Control Principle

Similarly to the Data Mangement function, this function is also done by the ActiveX control. Operating in the same manner as the Data Mangement function, this module allows users to place the supervisory controls on system's devices.

Users at both server and client sides can place commands on the system devices using the Internet browser, refered as the web interface. Upon receiving the commands, the DS modules will perform the simulation task and give back to the interface the responses corresponding to the commands

❖ In this research, because we are going to use the simulated device, an interface is also provided on the DS module. From the DS interface, the system operator can actually operate the DS module directly as if he or she is operating on the real device.

- **Device Management**

Using IP addresses, the two following functions can be added:

- Add/Remove: DS modules can be added or removed based on the IP address of the corresponding computers.
- Search: This function allows seeing which devices are available. Statuses of system's devices are shown afterwards.

The search job is done by the server, client users only need to insert the IP range of the LAN where DSs are installed and press the **Search** button. Pressing this button means that a request has been sent to the server, the server will proceed with the search steps and returns back the search results to the corresponding client.

3. Event Log

Event Log is a common tool in most distributed system since it is required for the system administrator to know and monitor any event occurred in the system.
Normally, typical event log tools allow the users to collect, analyze, archive and report on any event or action happened in the system. There can be many types of logs and log tools in a distributed system as listed in Table 4.1 below:

Table 4.1 Event Log Type

Application and Server Status Monitoring	Notification Methods
+ Cluster Monitor	+ Alerts
+ Event Alarm	+ Command Script
+ Exchange Monitor	+ Forward to
+ File Monitor	+ Instant Messenger
+ IIS Monitor	+ Pagers
+ Performance Alarm	+ Short Message
+ Process Monitor	+ Email
+ Service Monitor	+ Web Post
+ SQL Monitor	+ Beeps
Internet Service Monitoring	+ Net Send
+ FTP Monitor	+ Sound Files
+ TCP Port Monitor	+ Text-to-Speech
+ Ping Monitor	Data Collector and Real-Time Monitors
+ POP3 Monitor	+ Event Collector
+ SMTP Monitor	+ Event File Collector
+ SNMP Monitor	+ Inventory Collector
+ Web Page Monitor	+ Performance Collector

The following features can also be achieved with Event Log:

- Monitor event log, application log, network devices.
- Keep track of system performance, software inventories, and transactions.
- Baseline and leverage system performance, perform capacity planning, see bottlenecks and immediately recognize problems
- Make in various formats and print out reports.

4. Security Management

As proposed in Chapter 3, the security issue of the system is considered on:

1. **Router Level** → 2. **Server Application**

1. Router Level

This is the hardware's function which is already integrated into the router itself. This feature is common for most types of routers. It refers to the IP configuration of the router about the firewall rules that allow or that do not allow a specific type of connection trying to connect to any computer located behind the router.

The server wherein the application software is hosted lies behind the router. Therefore we can use the router's firewall rules as the first barrier that any kind of connection to the server computer has to pass.

2. Server Application

There will be three tools in the server's software.

- User Authentication
- IP Approval
- User Permission

User Authentication authenticates existing users with username and password. This tool allows the administrator to lock or unlock the system acces of a specific user. User Permission is used to specify the actions that a user can perform on the system and IP Approval to authenticate which client computers can gain access to the server.

C. Device Simulator

With Internet-based solution for SCADA systems, we can control and manage our processes or devices via Wide Area Network (WAN) or Internet network. Control commands will be sent to the devices by system's operators via TCP/IP protocol, and then the devices' data or data from the processes will be sent back to the SCADA interface in the same manner using TCP/IP protocol.

With this idea, in order to control the devices, the devices themselves have to support the TCP/IP or they must be connected to an additional module which supports the TCP/IP protocol. In this research, we assume having the presence of the Ethernet IO board with a separately excited DC motor and this motor is managed by using the IP addresses of the Ethernet IO boards to which it is connected.

However, for the reason of test time saving, all the work and operation regarding system devices are simulated into software modules. These modules, as mentioned, are called Device Simulator (DS). DS is a software application written in Visual C# and Matlab. DS will completely simulate the DC motor as if the motor has been connected to the Ethernet IO boards and in real operation. There will be one DS module corresponding to the DC motor.

***Working frame of the DS is described below*:**

- DS will be installed in a separate computer equipped with an IP addresses. This computer acts as the Ethernet IO board, and it is connected in the same Ethernet LAN together with the server computer.
- The web-based HMI software embedded with the ActiveX control provides users the ability to communicate with the DS modules via TCP/IP protocol.
- The DS module also contains an interface which has the same data display and control buttons as the main web-based interface. This lets the system's operator at the device side has the same look as if they are sitting at the server computer or at a remotely authenticated client computer.

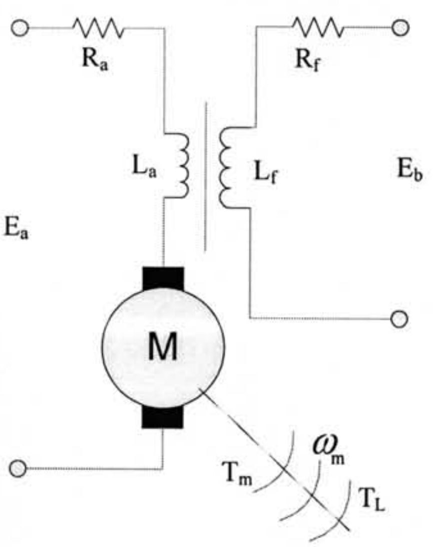
In the following parts, the operational principle of the DS motor will be given, followed by the details of the DS implementation.

1. How a Separately Excited DC Motors work

DC motors consist of one set of coils, called an armature, inside another set of coils or a set of permanent magnets, called the stator. Applying a voltage to the coils produces a torque in the armature, resulting in motion.

See Figures 4.13 and 4.14 for the physical and mathematical models of DC motors.

- Physical Model:



Specifications:

- R_f & L_f: Field Resistance & Inductance
- R_a & L_a: Armature Resistance & Inductance
- J_m & B_m: Rotor Inertia & Viscous Frictional Coefficient
- K_i & K_b: Torque and Back-Emf Constant
- E_a: Armature Voltage
- T_L: Load Torque
- E_b: Induced Voltage

Figure 4.13 Physical Model of a DC Motor

- Mathematical Model:

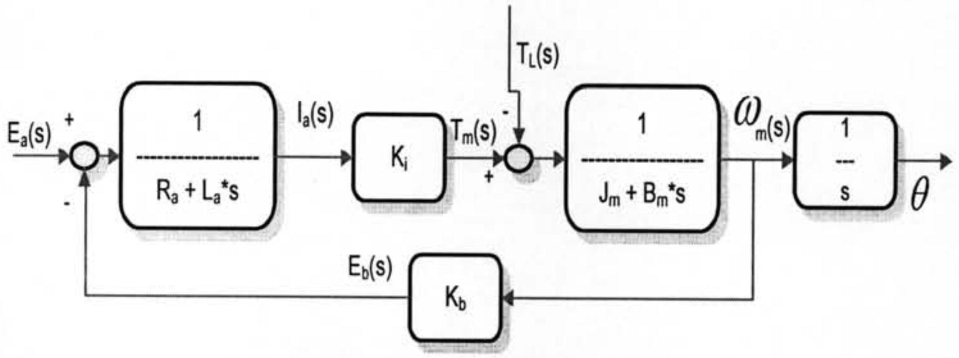


Figure 4.14 Mathematical Model of a DC Motor

- Detailed operation:

Torque and Current

Torque is proportional to the product of armature current and the resultant flux density per pole.

$T_m = K_i \times f \times I_a$

Where T_m is torque, K_i is torque constant, f is the flux density, and I_a is the armature current.

Speed, Voltage, and Induced Voltage

Resistance of the armature widings has only a minor effect on armature current. Current is mostly determined by the voltage induced in the windings by their movement through the field. This induced voltage, also called "back-emf" is opposite in polarity to the applied voltage, and serves to decrease the effective value of that voltage, and thereby decreases the current in the armature. An increase in voltage will result in an increase in armature current, producing an increase in torque, and acceleration. As speed increases, induced voltage will increase, causing current and torque to decrease, until torque again equals the load or induced voltage equals the applied voltage. A decrease in voltage will result in a decrease of armature current, and a decrease in torque, causing the motor to slow down. Induced voltage may momentarily be higher than the applied voltage, causing the motor to act as a generator. This is the essense of regenerative breaking.

Induced voltage is proportional to speed and field strength.

$$\mathbf{E_b = K_b * \omega_m * f}$$

Where E_b is induced voltage, K_b is "back-emf" constant particular to the motor, ω_m is the speed of the motor, and f is the field strength.

This can be solved for speed to get the "Speed Equation" for a motor:

$$\omega_m = \frac{E_b}{K_b * f}$$

The speed of the motor is inversely proportional to field strength. This means, as field strength decreases, speed increases.

2. DS Implementation

The DS module, which is going to perform all the functions as stated above, will be built with the four following sub-modules:

Configuration, **Command Service**, **Command Handler**, and **Result Distribution**.

- **Functional Diagram**

Figure 4.15 shows the four sub-modules in the entire DS module. Other parts are external interfaces which the DS module has to work with.

- **Functional Description**

o **Configuration**

- Allows system's operator to insert the general information about the motor such as the motor's name, type, manufacturer, model ID, etc.

- Allows system's operator to insert the specification values, which are mentioned earlier, of the motors. These values depend on the motor itself and will never be changed during the motor's operation. These values are quoted from the motor catalog and fixed into the DS module.

All of these data are to be stored in the DS's memory for further use in calculating the reaction of the motor and for device management purpose.

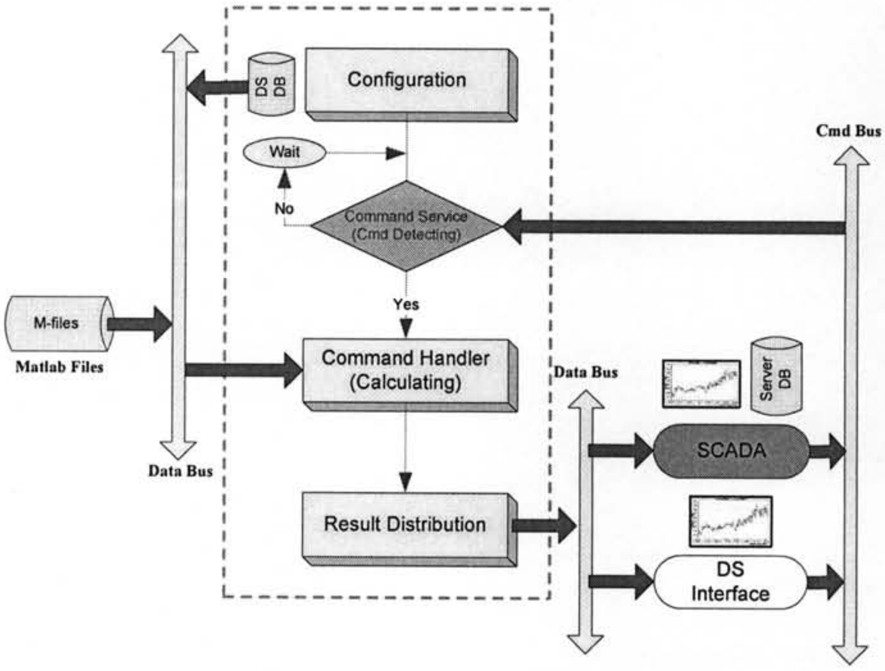


Figure 4.15 DS Operational and Interface Diagram

- **Command Service**

In most cases, this sub-module receives orders (commands) from users via TCP/IP protocol and transfers these commands to the sub-module Command Handler to process. Besides, users can also directly give commands to DS from the DS interface. For each Command Service sub-module, apart from the IP address of the corresponding computer where DS is installed, we have to specify a port number opened in the DS computer to make it able to receive the command. In other words, for any entity located in the Internet network, it must be addressed by an IP and a port number for any desired communication via TCP/IP protocol. Here, we need to do the same thing for the Command Service sub-module. We can choose any number for the port in range of 0 and 65534, but try not to use the port of any common service. In this thesis, the number 4206 is reserved for the stated purpose.

In case the command is given directly from the DS interface, it is transferred directly to the Command Handler module for further processing.

- **Command Handler**

Upon receiving a new command, the DS module will call the corresponding Matlab m-file function. This m-file has the task of performing the calculation which corresponds to a new set of parameters received from the Command Service sub-module. Among the parameters sent by the Command Service sub-module, the motor's specifications are sent only one time and they are kept constant during the motor's operation and in any further calculation. These parameters are stored in the DS's memory and are sent to the Configuration sub-module for only one time. For next processing times, only the data regarding the commands from users are to be sent to the Command Handler sub-module.

Once the calculation has been done, this sub-module outputs a new set of data.

In this research, the DS module supports 4 operation schemes which are:

Turn ON Device, Turn off Device, WithoutPID, and **WithPID.**

- **Turn ON Device**:

This command activates the DS module to run. The DS module is ready to receive additional commands from system users

- **Turn off Device**:

This command stops the DS module.

- **WithoutPID**:

The DS must be in the ON state. In this operation, the DS returns the operation result in the case the DC motor is working without any effect from controllers. Users can control the DS module by changing the armature voltage E_a and the load torque T_L. Changes in these two values will result in the change of the DC motor's reaction.

- **WithPID**:

The DS must also be in the ON state. In this operation scheme, a controller will be introduced into the DC motor to achieve a better performance and bring the controllability of the motors. There are many control algorithms which can be applied on DC motors. In this research, the Propotional-Integral-Derivative (PID) controller will be used due to its popularity and simple implementation.

The PID control algorithm is used to make the motor's output conform to the set value at which users want it to be.

The block diagram of a typical PID controller is presented in Figure 4.16.

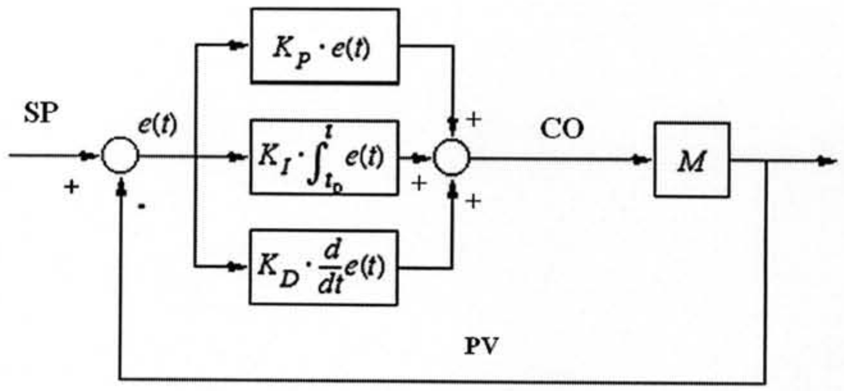

SP
e(t)
$K_p \cdot e(t)$
$K_I \cdot \int_{t_0}^{t} e(t)$
$K_D \cdot \frac{d}{dt} e(t)$
CO
M
PV
+
-
+
+
+

Figure 4.16 PID Controller Block Diagram

From the diagram, we have the output of the PID controller is calculated as:

$$CO = K_P * e(t) + K_i * \int e(t) * dt + K_d * \frac{de(t)}{dt}$$

Where: CO: Controller Output

e = SP-PV, error.

SP: Setpoint

PV: Process Variable

Generally, PID controller is a feedback control loop mechanism. The PID controller attempts to correct the error between a measured process variable and a desired setpoint by calculating and then outputting a corrective action that can adjust the process accordingly. The PID controller calculation (algorithm) involves three separate modes; the Proportional mode, the Integral mode and Derivative mode. The Proportional mode determines the reaction to the current error, the Integral mode determines the reaction based on recent errors and the Derivative mode determines the reaction based on the rate by which the error has been changing. The weighted sum of the three modes is outputted as a corrective action to a control element such as a control valve or heating element.

- *Proportional mode*:

Proportional mode responds to a change in the process variable proportional to the measured current error value. The proportional response can be adjusted by multiplying the error by a constant K_p, called the proportional gain (or constant).

- *Integral mode*:

With integral mode, the controller output is proportional to the amount and duration of the error signal. The integral mode algorithm calculates the accumulated proportional offset over time that should have been corrected previously (finding the offset's *integral*). While this will force the controller to approach the setpoint quicker than a proportional controller alone and eliminate steady state error, it also contributes to system instability as the controller will always be responding to past values. This instability causes the process to overshoot the setpoint since the integral value will continue to be added to the output value; even after the process variable has reached the desired setpoint. The responsiveness of the integral function can be calibrated to the specific process by adjusting the constant T_i, called the integral time.

$$K_i = \frac{K_p}{T_i}$$

- *Derivative mode*:

Derivative mode acts as a braking action to the controller response as the process variable approaches the setpoint. To accomplish this the predicted process error at a time in the future T_d is calculated by analyzing the slope of error versus time or the rate of error change (the first derivative of the error) and adding the anticipated proportional mode response to the current correction. Derivative control is used to reduce the magnitude of the overshoot produced by the integral component, but the controller will be a bit slower to reach the setpoint initially. By adjusting the constant, T_d, the derivative time, the braking action sensitivity is controlled.

$$K_d = K_p * T_d$$

With this operating scheme, users are provided with the ability to control the simulated DC motor by changing the Setpoint, which is the desired speed, and the parameters of the PID controller. The mathematical model for the DC motor with the PID controller integrated in is illustrated in the Figure 4.17.

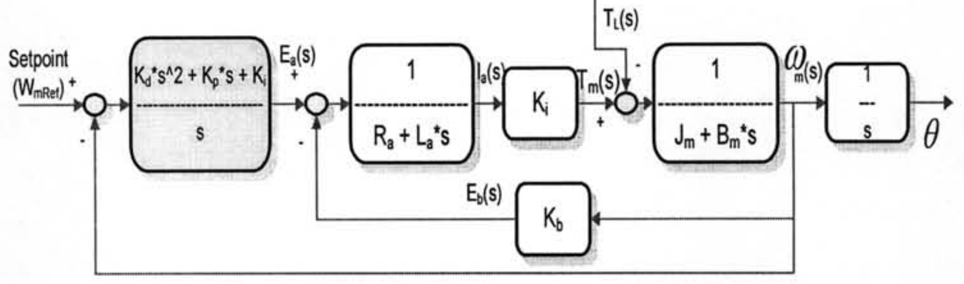


Figure 4.17 DC Motor with PID Controller Block Diagram

Where: W_{mRef} is the desired speed at which users want the DS to run

The designer, when designing the system, will choose an optimum set of the controller parameters which are K_p, K_i, K_d. These values are displayed on the SCADA interface when the system is working on this scheme. Users normally are not allowed to change these values. However, in the simulation scheme of this research, users can change these values either from the SCADA interface or on the DS interface. This is to visualize the effect of the PID controller itself, which is good for the study purpose. Under this operation scheme, system users interact with the simulated motor by changing the reference speed to whichever value and the response speed of the motor will conform and be stable to that value.

**For real motors, the reference speed is limited due to the physical limitations. However, with the software simulation, this speed can be increased to any value.*

Different reponses of the motor under different PID parameters can be viewed by changing the values of K_p, K_i, K_d.

- **Result Distribution**

After the sub-module Command Handler finishes processing the calculation, it generates a new data set about the motor's reaction. This sub-module will then take these data as its input and perform the two following actions:

- Send these data to the server computer to be stored in the DB and displayed on the SCADA interface. This allows the remote users to view and evaluate the data.

In this case, this sub-module acts as a communication port to export the data to the SCADA interface. Upon receiving a request, this sub-module opens the specified port of the computer where DS is installed. A transparent path between the server and the DS module is created, data values will then be sent to the server every 0.25 second.

- Display these data on the DS's interface.

Data values displayed here are exactly the same as the values shown on the main SCADA interface at the server computer and at client computers. However, from the communication mechanism, we will see a slight time difference between data at the DS screen and data at the client interface, since data have to travel over the Internet network from the DS module to the client interface. How much this difference is will be presented in the experimental results coming in the next part of this research.

4.3 Testing Results

4.3.1 Getting Started

This part presents the detailed operation of the system. All the operational screen capturings are done at a client computer to show the communication over the Internet.

In the server computer, the SCADA folder containing all the files of the system application software is stored in **C:\Inetpub\wwwroot**. This is the default path for all web application softwares stored in a server computer.

Normally, the administrator will work at the server computer. However, if a user is granted the admin role, this user can also act as the administrator and be able to remotely manage the system.

To enable the remote access to the server website, a process called *domain name registration* must be done. Usually, to access to a computer, we need to know its IP address. However, the IP address of any computer in the Internet might be changed plus the fact that it is really hard for users to remember a set of numbers representing the IP address of the computer. Therefore, a memorable name is necessary for any web-based application. The Domain Name System or Service (DNS*) is a process that maps hostnames which is how we address servers by names, to IP addresses, which is how computers address servers by a set of numbers. In other words, the DNS system acts as the phonebook for the Internet.

**How DNS works:*

The domain name space consists of a tree of domain names. Each node or leaf in the tree has one or more resource records, which hold information associated with the domain name. The tree sub-divides into zones. A zone consists of a collection of connected nodes authoritatively served by an authoritative DNS name server. A single name server can host several zones.

A resolver looks up the information associated with nodes. A resolver knows how to communicate with name servers by sending DNS requests and receiving DNS responses. Resolving usually entails iterating through several name servers to find the needed information. Some resolvers function simplistically and can only communicate with a single name server. These simple resolvers rely on a name server to perform the work of finding information for them.

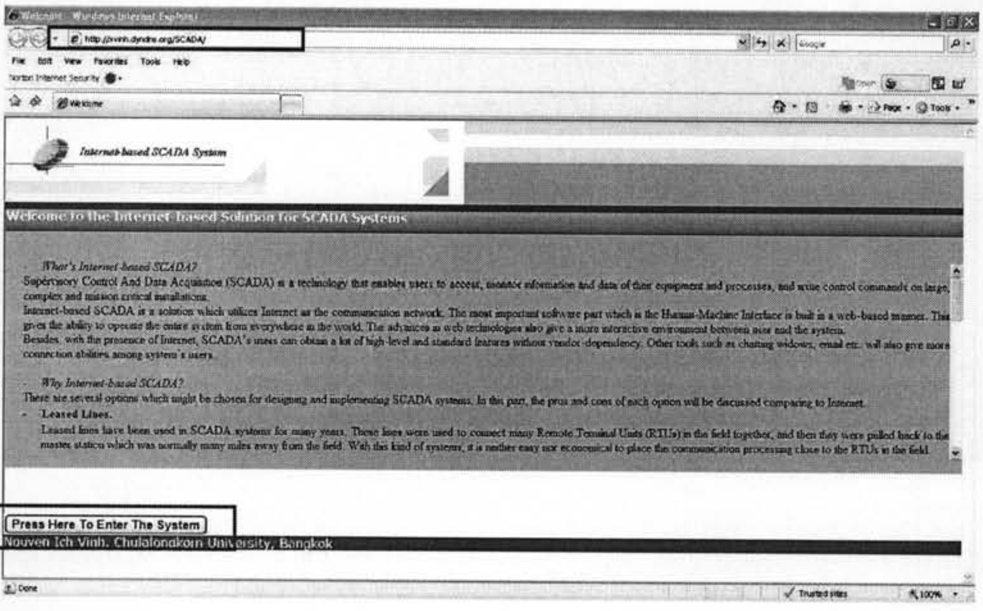
Underneath each domain name is an IP address. For example, when we visited the website www.dyndns.org, the DNS system resolved that hostname as 63.208.196.100

which is the IP address of the server where the website is stored. This translation happens every time we access a website.

There can be many DNS organizations to register the DNS service. One of these is DynDns, and in this research I use DynDns because it allows registering for free. However, with the free service, we can only use some specific hostnames which have been allowed by DynDns, and they have not been registered by other DynDns users. In this thesis demonstration, I have chosen the name **xvinh.dyndns.org**.

- To start working on the system at the server computer, users need to place the path: http://localhost/SCADA/ into the address field in the Internet browser.
- To start working on the system at the client computer, users need to place the path: http://xvinh.dyndns.org/SCADA/

The Welcome page is then opened as in Figure 4.18.


http://xvinh.dyndns.org/SCADA/
File Edit View Favorites Tools Help
Welcome
Internet-based SCADA System
Press Here To Enter The System
Done

Welcome to the Internet-based Solution for SCADA Systems

- *What's Internet-based SCADA?*

Supervisory Control And Data Acquisition (SCADA) is a technology that enables users to access, monitor information and data of their equipment and processes, and write control commands on large, complex and mission critical installations.
Internet-based SCADA is a solution which utilizes Internet as the communication network. The most important software part which is the Human-Machine Interface is built in a web-based manner. This gives the ability to operate the entire system from everywhere in the world. The advances in web technologies also give a more interactive environment between user and the system.
Besides, with the presence of Internet, SCADA's users can obtain a lot of high-level and standard features without vendor-dependency. Other tools such as chatting widows, email etc. will also give more connection abilities among system's users.

- *Why Internet-based SCADA?*

There are several options which might be chosen for designing and implementing SCADA systems. In this part, the pros and cons of each option will be discussed comparing to Internet.

- **Leased Lines.**

  Leased lines have been used in SCADA systems for many years. These lines were used to connect many Remote Terminal Units (RTUs) in the field together, and then they were pulled back to the master station which was normally many miles away from the field. With this kind of systems, it is neither easy nor economical to place the communication processing close to the RTUs in the field.

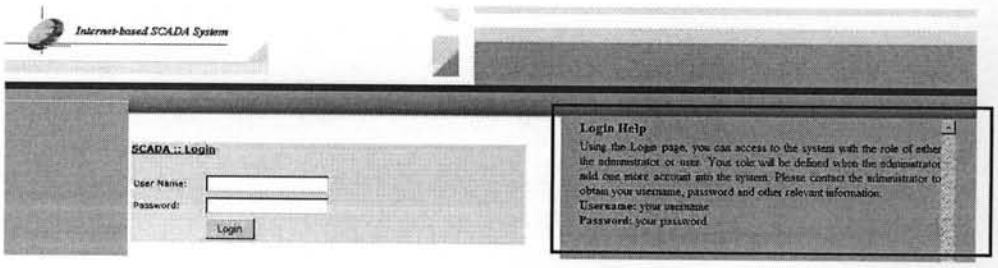
Nguyen Ich Vinh, Chulalongkorn University, Bangkok

Figure 4.18 Welcome Page

This page displays the overview information about the Internet-based solution for SCADA systems. A definition and a comparision among different options in implementing SCADA systems are presented here.

To enter the system, users need to press the button **Press Here To Enter The System** given as shown in Figure 4.18.

The Login page is opened when this button has been pressed as in Figure 4.19.



SCADA :: Login

User Name:
Password:

Login Help

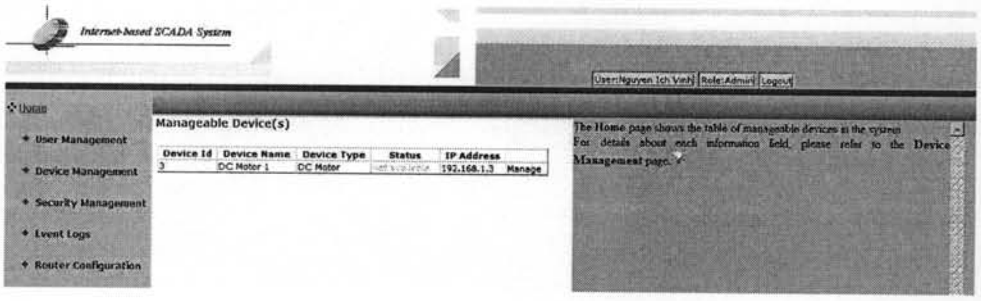
Using the Login page, you can access to the system with the role of either the administrator or user. Your role will be defined when the administrator add one more account into the system. Please contact the administrator to obtain your username, password and other relevant information.

Username: your username

Password: your password

Figure 4.19 Login Page

When a user is successfully logged in, the Home page is opened taking user direcly to the lisf of manageable devices as shown in Figure 4.20.



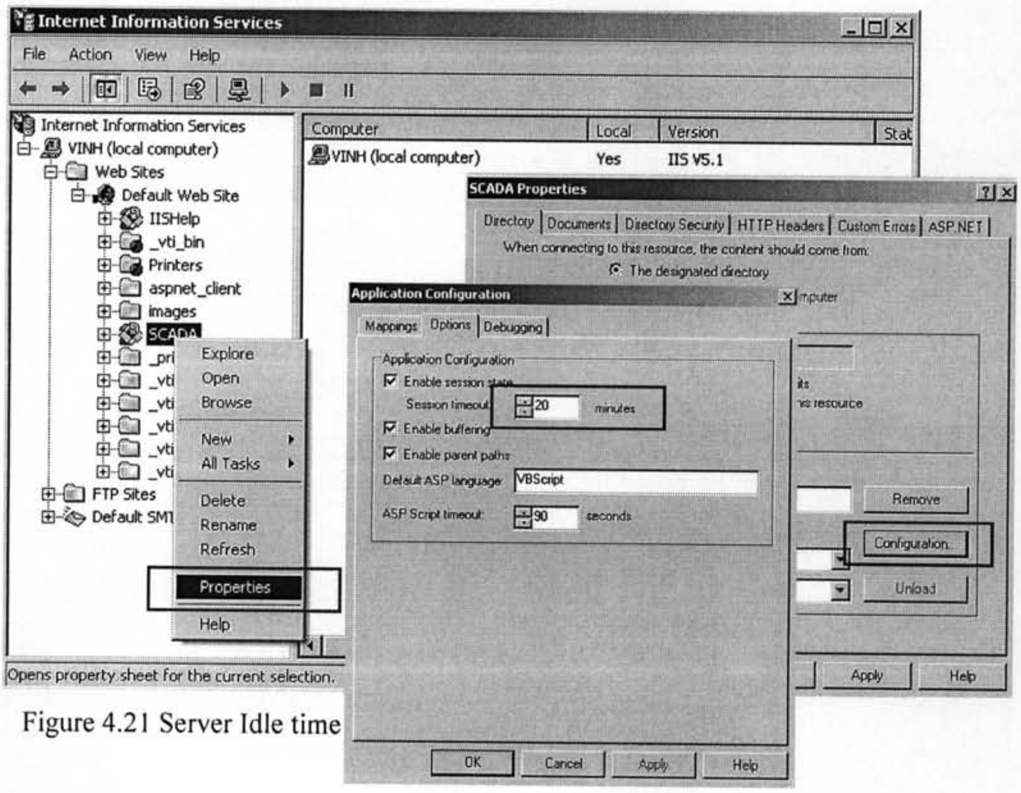
User:Nguyen Ich Vinh Role:Admin

Manageable Device(s)

Device Id	Device Name	Device Type	Status	IP Address	
3	DC Motor 1	DC Motor	[illegible]	192.168.1.3	Manage

The Home page shows the table of manageable devices in the system. For details about each information field, please refer to the Device Management page.

Figure 4.20 Home Page



VINH (local computer) Yes IIS V5.1

When connecting to this resource, the content should come from:
The designated directory

Application Configuration
☑ Enable session state
Session timeout: 20 minutes
☑ Enable buffering
☑ Enable parent paths
Default ASP language: VBScript
ASP Script timeout: 90 seconds

Opens property sheet for the current selection.

Figure 4.21 Server Idle time

From the Homepage, links are provided for users to go to every function of the system. Details are presented in the following parts in this chapter.

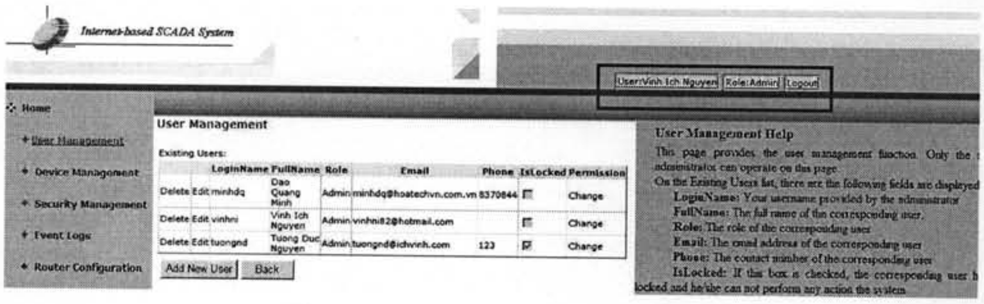
** Link to each function of the system will open a new operation page. Note that, if users leave an operation page for more than 20 minutes, meaning that the system is idle for more than 20 minutes, then when users move to a new page, the system will ask for logging in again. This is the default setting of the IIS system and is recommended for security purpose in case the operating computer is left unattended. This period of time might be changed by going to the IIS setting for our application software as shown in Figure 4.21.*

On each operation page, the corresponding operational information is displayed on the right-hand side of the page, as red-marked in Figure 4.19.

4.3.2 Detailed Operations

In the following content, the actual operation of a system under the control of the administrator together with all the screenshots of system interfaces is presented.

1. User Management



User Management

Existing Users:

		LoginName	FullName	Role	Email	Phone	IsLocked	Permission
Delete	Edit	minhdq	Dao Quang Minh	Admin	minhdq@hoatechvn.com.vn	8370844	☐	Change
Delete	Edit	vinhni	Vinh Ich Nguyen	Admin	vinhni82@hotmail.com		☐	Change
Delete	Edit	tuongnd	Tuong Duc Nguyen	Admin	tuongnd@ichvinh.com	123	☑	Change

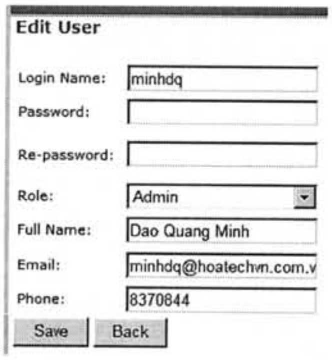
User Management Help

This page provides the user management function. Only the administrator can operate on this page.
On the Existing Users list, there are the following fields are displayed
LoginName: Your username provided by the administrator
FullName: The full name of the corresponding user.
Role: The role of the corresponding user
Email: The email address of the corresponding user
Phone: The contact number of the corresponding user
IsLocked: If this box is checked, the corresponding user h locked and he/she can not perform any action the system

Figure 4.22 User Management Page

Figure 4.22 is the screenshot of the User Management inferface. The active user is Vinh Ich Nguyen with the administrator role on the system. All the managemen functions on the left-hand side of the page are available for this user.

With a particular user, to delete, simply click on the **Delete** button; the **Edit** button is provided to edit the profile information. Note that the username once created cannot be changed by users, but the profile information. If the administrator wants to change the username of a specific user, the administrator has to delete that user and add a new user with the new username. See Figure 4.23 for Edit User operation.



Edit User

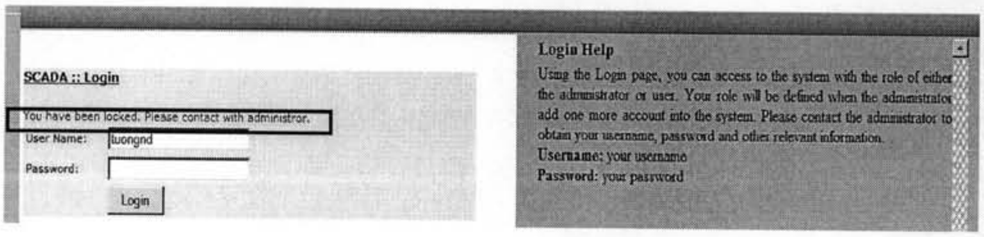
Login Name:	minhdq
Password:	
Re-password:	
Role:	Admin
Full Name:	Dao Quang Minh
Email:	minhdq@hoatechvn.com.v
Phone:	8370844

When the **Edit** button is pressed, the **Edit User** page is opened and the administrator can re-enter all the information belonging to that particular user.

Figure 4.23 shows the information of user Dao Quang Minh being edited. Changes made are effective when the **Save** button is pressed.

Figure 4.23 Edit User

The button **Change** is used to alter the specific rights of each user on the system. The role of each user on the system is displayed in the Role field. Whether a user is locked is also displayed in the IsLocked field. For example, the user Tuong Duc Nguyen with Login name *tuongnd* is locked and the status is displayed. If a user has been locked, an alert indicating that he or she has been locked will appear if that user tries to login the system as shown in Figure 4.24.



SCADA :: Login

You have been locked. Please contact with administror.

User Name:	tuongnd
Password:	

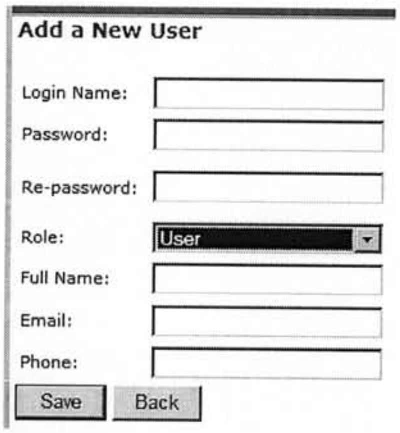
Login Help

Using the Login page, you can access to the system with the role of either the administrator or user. Your role will be defined when the administrator add one more account into the system. Please contact the administrator to obtain your username, password and other relevant information.

Username: your username

Password: your password

Figure 4.24 Locked Users



Add a New User

Login Name:	
Password:	
Re-password:	
Role:	User
Full Name:	
Email:	
Phone:	

The **Add New User** button opens the Add New User page, the administrator needs to insert all the profile information, then the Save button is to be pressed to save the user account to the system. The administrator will need to consider the role of the user and distribute the appropriate rights on the sytem. From now, the user added can perform actions which have been approved.

Figure 4.25 Add New User

On all the User Mangement pages, a **Back** button is provided for users to come back to the previous page.

2. Device and Data Management

As mentioned, in this research, ActiveX Control technology is used for the data and device management function. Details about activeX control are presented below with respect to web programming.

In general, ActiveX is an open integration platform that provides a way to create integrated programs and content for the Internet. Using ActiveX, we can insert multimedia effects, interactive objects, and sophisticated programs into a Web page. ActiveX is a standard that enables software components to interact with one another in a networked environment, regardless of the language(s) used to create them. Most World Wide Web (WWW) users utilize ActiveX technology in the form of ActiveX controls, ActiveX documents, and ActiveX scripts.

- *ActiveX control:*

ActiveX controls are components (or objects) that we can insert into a Web page or other program so that you can reuse packaged functionality that someone else programmed. For example, the ActiveX controls that are included with Internet Explorer enable us to enhance your Web pages with sophisticated formatting features and animation.

ActiveX controls can also be used in programs written in many programming languages and database languages.

- *ActiveX documents:*

When you are browsing with an ActiveX-aware Web browser, such as Internet Explorer, ActiveX documents enable us to open a program with its own toolbars and menus available. This means we can open non-HTML files, such as Microsoft Excel or Microsoft Word files, by using an ActiveX-aware Web browser.

- *ActiveX scripts:*

ActiveX scripts support most popular scripting languages, including Microsoft Visual Basic Script and JavaScript. ActiveX scripts can be used to integrate the behavior of several ActiveX controls or Java programs from the Web browser or server, extending their functionality.

Used in this research is the ActiveX control which, as mentioned, can be re-used within a computer or among computers in a network. An ActiveX control is written to handle all the data and device management functions. It is a software package which is written in VB and is migrated in the entire server software application. This ActiveX control can also be re-used in other web-based application.

- **Data Management**

System's data are generated from the operation of the DS modules. The Result Distribution sub-modules in the DS modules have the task of distributing the results to the SCADA interface and DS interfaces (See Figure 4.16 for more details).

On DS interfaces, it is quite straightforward for the data to be displayed. Once the data are generated from the Command Handler sub-modules, they will be sent to the DS graph to be displayed in real-time clocking.

The system application software are designed to ensure what happen in the DS module are reflected exactly the same on the SCADA pages. The ActiveX control sends requests to the DS modules asking for latest data in every 25ms. Once DS modules are turned on and in operation, they will acknowledge these requests. These data are to be stored in the server DB, and, on the other hand, they are also displayed in real-time clocking on the SCADA interface. In case DS modules are not operating, there will be no data sent back to the server and thus no data are to be recorded in the DB. From the client computer, when the Internet browser gets connected to the server's SCADA website, the ActiveX control is automatically downloaded and installed into the client computer. Once the **Turn ON Device** button is pressed, meaning that requests from client use are being sent to the server, the data in the server computer will be sent to the client interface to be displayed in real-time clocking. The ActiveX is programmed to send requests from client users to the server in 25ms periodically, which is the same period as the interval in which the server sends requests to the DS modules. Therefore, what is being displayed on the DS interface as well as the server SCADA pages will also be displayed on the client SCADA pages exactly the same.

Besides, the history data stored in the server's DB may be recalled at any time from the SCADA interface. Users can select any period of time until present.

Figures illustrating the Data Management function are listed in the next section in accordance with the Device Control function.

- **Data Transfer Speed and Time**

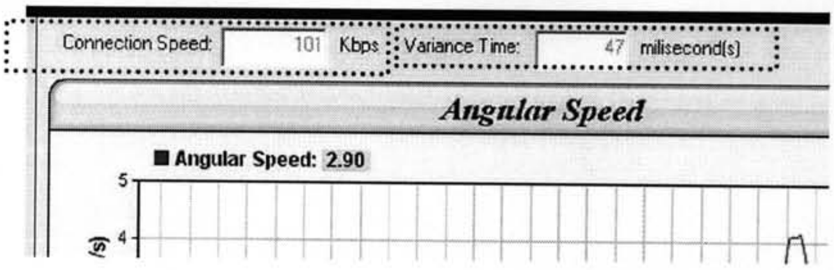
As dicussed in section B above, we need 3 parameters which are TBX, t_2 and t_1.

- TBX is calculated by counting the number of bytes in one time exchanging request and response between server and client.

- t_2 and t_1 are calculated on the client computer by the ActiveX control

Notice that these values are only the estimated values. The length of the http request and response also varies depending on the type of the http messages to be used.

As calculated, in this testing system, the TBX value is 250 bytes approximately. Figure 4.26 shows the real values of the Data Transfer Speed and Varicance Time displayed on the ActiveX control on the SCADA interface.



Connection Speed: 101 Kbps

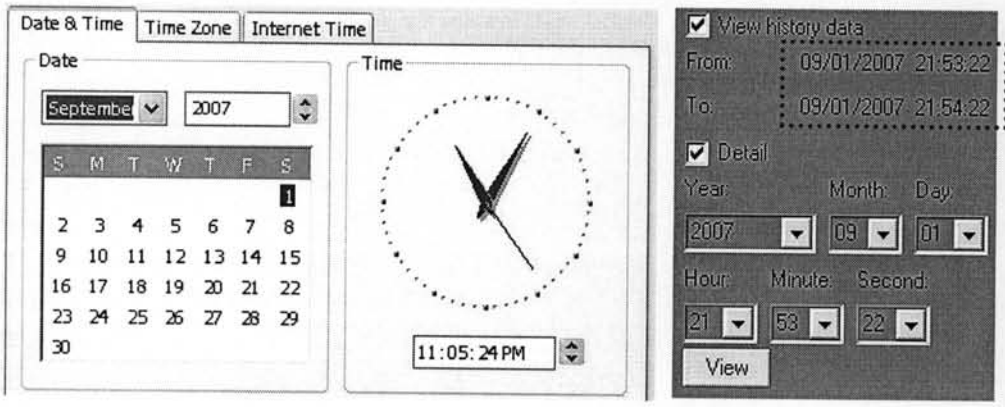
Variance Time: 47 milisecond(s)

Figure 4.26 Connection Speed and Variance Time

From Figure 4.26, if we take 100 kbps as the Data Transfer Speed and 50 ms as the Variance Time, we will have TBX = $100 * 10^3 * 50 * 10^{-3} * 2^{-1}$ = **250** Bytes.

- **History Data**

  Figure 4.27 shows the system date and time and the chosen period



Date & Time | Time Zone | Internet Time

Date: September 2007

S	M	T	W	T	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						

Time: 11:05:24 PM

[✓] View history data

From: 09/01/2007 21:53:22

To: 09/01/2007 21:54:22

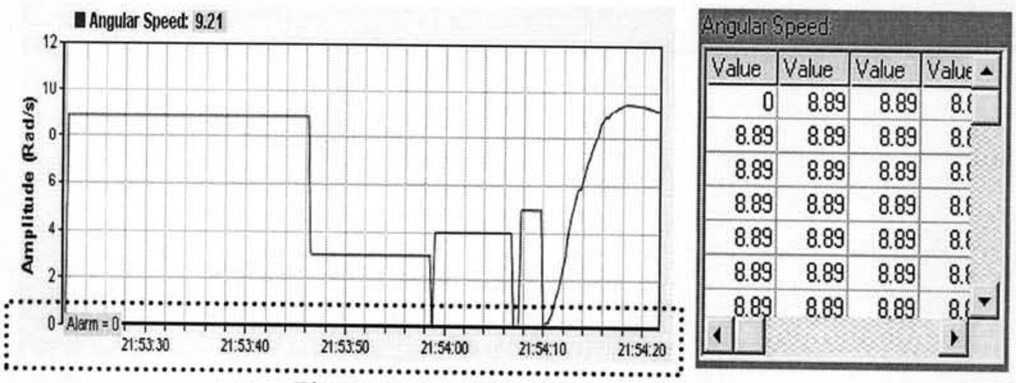
[✓] Detail

Year: 2007 Month: 09 Day: 01

Hour: 21 Minute: 53 Second: 22

View

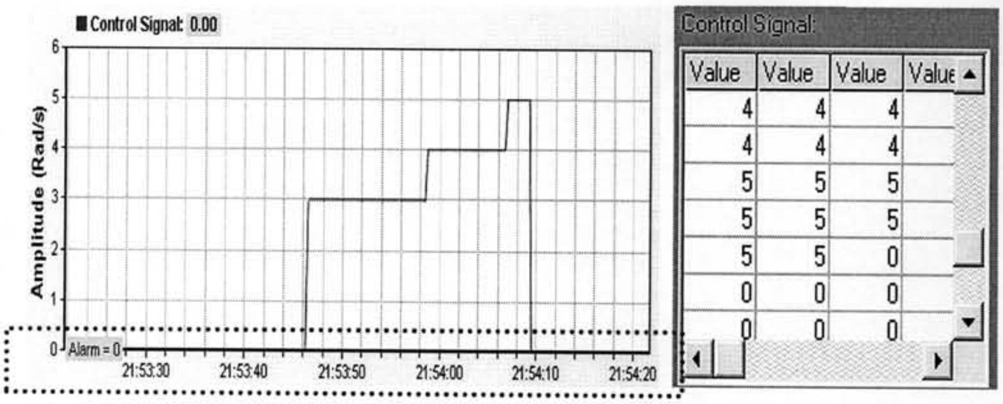
Figure 4.27 System Time and History Time



Angular Speed

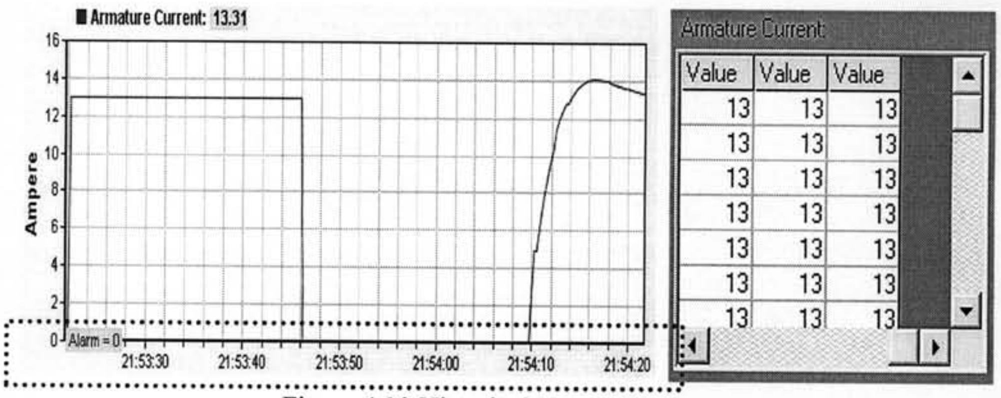
Value	Value	Value	Value
0	8.89	8.89	8.
8.89	8.89	8.89	8.
8.89	8.89	8.89	8.
8.89	8.89	8.89	8.
8.89	8.89	8.89	8.
8.89	8.89	8.89	8.
8.89	8.89	8.89	8.

Figure 4.28 Historical Angular Speed



Value	Value	Value	Valu
4	4	4	
4	4	4	
5	5	5	
5	5	5	
5	5	0	
0	0	0	
0	0	0	

Figure 4.29 Historical Control Signal



Value	Value	Value
13	13	13
13	13	13
13	13	13
13	13	13
13	13	13
13	13	13
13	13	13

Figure 4.30 Historical Armature Current

- **Device Control**

As mentioned, the simulated DC motors are to be operated in two schemes which are free operation, **WithoutPID**, and with the PID controller, **WithPID**.

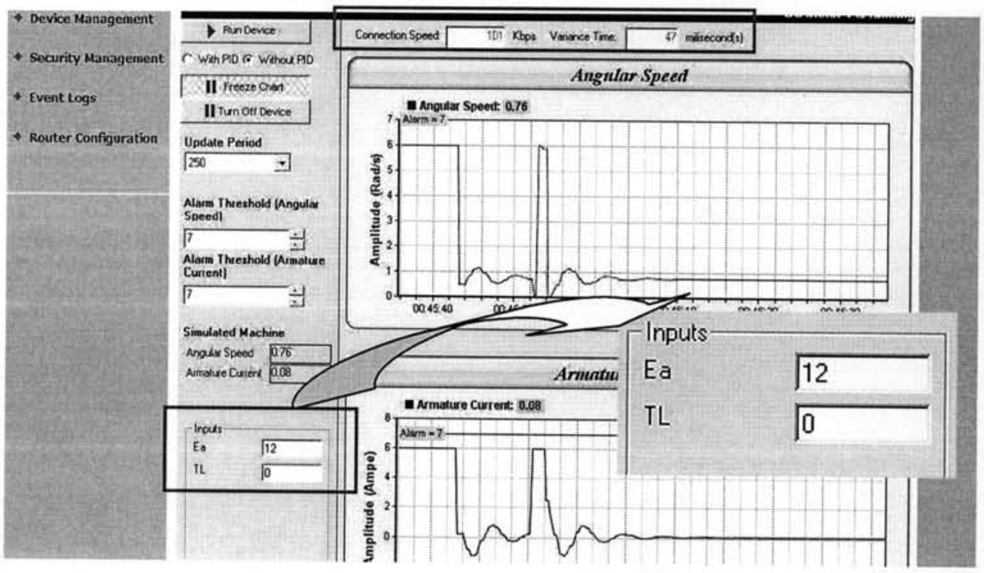
_WithoutPID:

The motor's operation is affected by the applied voltage E_a and the load torque T_L. Angular Speed and Armature Current of the motor are presented in Figure 4.31.

From this interface, users can make changes to the E_a and T_L values to see the different outputs of the simulated DC motors. However, in this case, the speed of the motor is unpredicted upon the changes in E_a and T_L.

_WithPID:

Under this scheme, no load is applied on the DC motor. The voltage being fed to the motor is the output of the PID controller.



Connection Speed: 101 Kbps Variance Time: 47 milisecond(s)

( ) With PID [✓ Without PID]

Update Period: 250

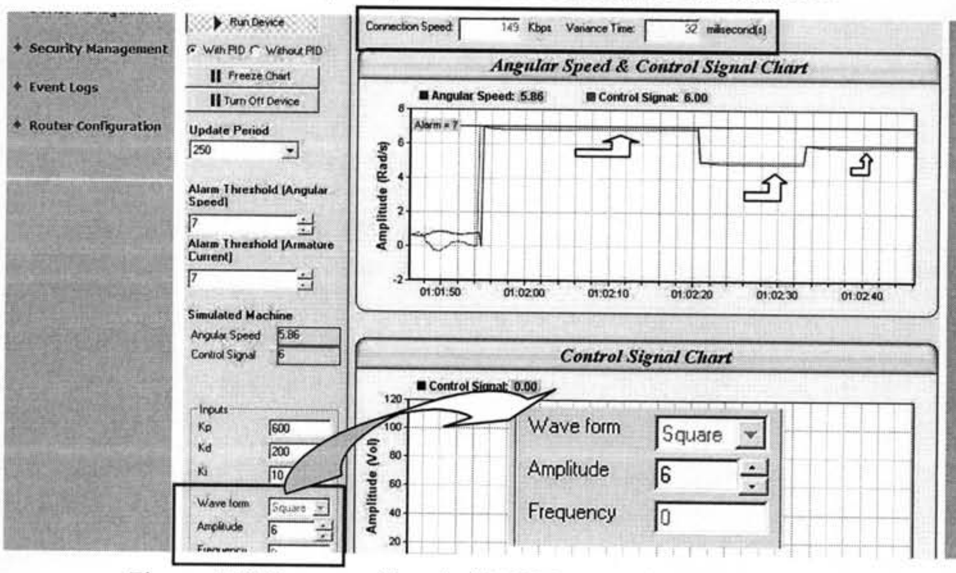
Alarm Threshold (Angular Speed): 7

Alarm Threshold (Armature Current): 7

Simulated Machine
Angular Speed: 0.76
Armature Current: 0.08

Inputs
Ea: 12
TL: 0

Figure 4.31 Step Response of DC Motor under no Controller



Connection Speed: 149 Kbps Variance Time: 32 milisecond(s)

[✓ With PID] ( ) Without PID

Update Period: 250

Alarm Threshold (Angular Speed): 7

Alarm Threshold (Armature Current): 7

Simulated Machine
Angular Speed: 5.86
Control Signal: 6

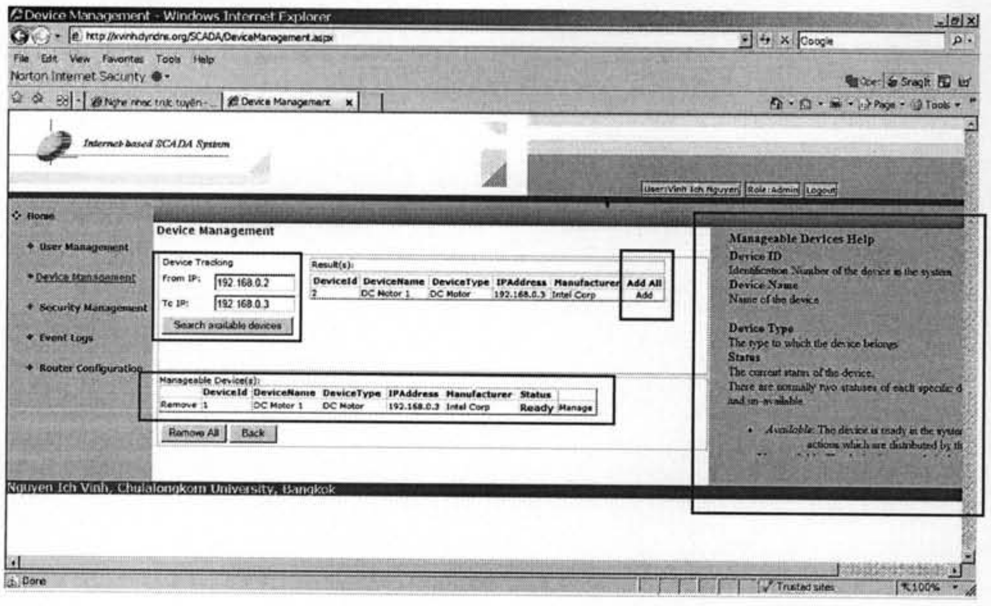
Inputs
Kp: 600
Kd: 200
Ki: 10

Wave form: Square
Amplitude: 6
Frequency: 0

Figure 4.32 Response Speed of DC Motor under PID Controller

With the PID controller's presence, users can regulate the speed of the motor to any value, called the reference speed, and the speed of the motor will conform to this reference speed. In Figure 4.32, we can see the response speed of the motor running stably after a very short transitional time. The speed of the motor is regulated from 7 rad/s to 5 rad/s and then is increased to 6 rad/s stably. Changes in speed are made by changing the amplitude of control signal. The green line represents the control signal and the red line represents the speed of the motor.

- **Device Management**



Device Management

Device Tracking

From IP: 192.168.0.2

To IP: 192.168.0.3

Result(s):

DeviceId	DeviceName	DeviceType	IPAddress	Manufacturer	Add All
2	DC Motor 1	DC Motor	192.168.0.3	Intel Corp	Add

Manageable Device(s):

	DeviceId	DeviceName	DeviceType	IPAddress	Manufacturer	Status	
Remove	1	DC Motor 1	DC Motor	192.168.0.3	Intel Corp	Ready	Manage

Manageable Devices Help

Device ID

Identification Number of the device in the system

Device Name

Name of the device

Device Type

The type to which the device belongs

Status

The current status of the device.

There are normally two statuses of each specific d
and un-available.

- *Available:* The device is ready in the syster
actions which are distributed by th

Figure 4.33 Device Management Page

As mentioned, system users have the ability to track which devices are being attached to the system, add the found devices to the manageable device list as well as remove devices out of this list for such purposes as repair, maintenance, etc. Figure 4.33 shows the following tools which can be applied onto the system's devices. Information regarding the devices is displayed on the right-hand side of the page.

- *Device Track:*

Users need to insert the IP range of the LAN network, to which the server computer and the computers where DS modules are installed, into the **From IP** and **To IP** fields. When the button **Search available devices** is pressed, system will return the found devices.

- *Device Add:*

The **Add** button next to a device once pressed will add the device into the device manageable list. If users add a device which is already in the list, the system will show an alert saying that the device has already been in the list.
In case users want to add all the found devices into the manageable device list for further works, the **Add All** button is provided.

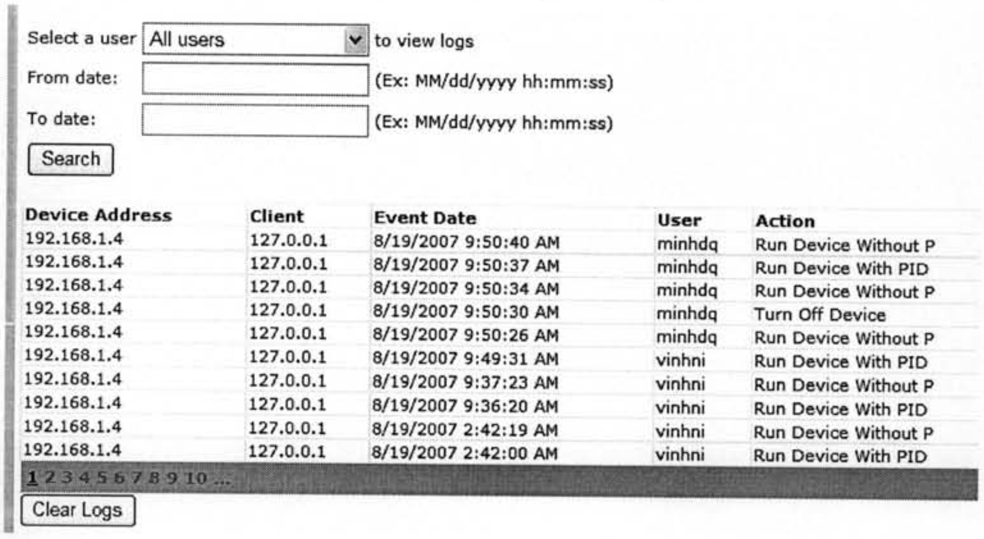
- *Device Remove:*

In some cases, some devices need to be isolated from the system for the need of physical maintenance or newly replacing. In such cases, users can islolate the intended devices by pressing the corresponding **Remove** buttons on the left-hand most column in the list, and the device will be removed out of the system's control. To remove all the devices out of the manageable device list, the button **Remove All** is provided. Changes made here also affect the Home page.

** The Device Management page and the Home page automatically refresh every 5 seconds to retrieve the latest statuses of the system's devices.*

3. Event Log

In this application, the event log function is restricted at the action log level. Every control command placed on system devices will be recorded and then inserted into the the scada_eventlogs table in the DB. By going to the **Event Logs** link on the SCADA pages, we also have all events displayed (See Figure 4.34).



Device Address	Client	Event Date	User	Action
192.168.1.4	127.0.0.1	8/19/2007 9:50:40 AM	minhdq	Run Device Without P
192.168.1.4	127.0.0.1	8/19/2007 9:50:37 AM	minhdq	Run Device With PID
192.168.1.4	127.0.0.1	8/19/2007 9:50:34 AM	minhdq	Run Device Without P
192.168.1.4	127.0.0.1	8/19/2007 9:50:30 AM	minhdq	Turn Off Device
192.168.1.4	127.0.0.1	8/19/2007 9:50:26 AM	minhdq	Run Device Without P
192.168.1.4	127.0.0.1	8/19/2007 9:49:31 AM	vinhni	Run Device With PID
192.168.1.4	127.0.0.1	8/19/2007 9:37:23 AM	vinhni	Run Device Without P
192.168.1.4	127.0.0.1	8/19/2007 9:36:20 AM	vinhni	Run Device With PID
192.168.1.4	127.0.0.1	8/19/2007 2:42:19 AM	vinhni	Run Device Without P
192.168.1.4	127.0.0.1	8/19/2007 2:42:00 AM	vinhni	Run Device With PID

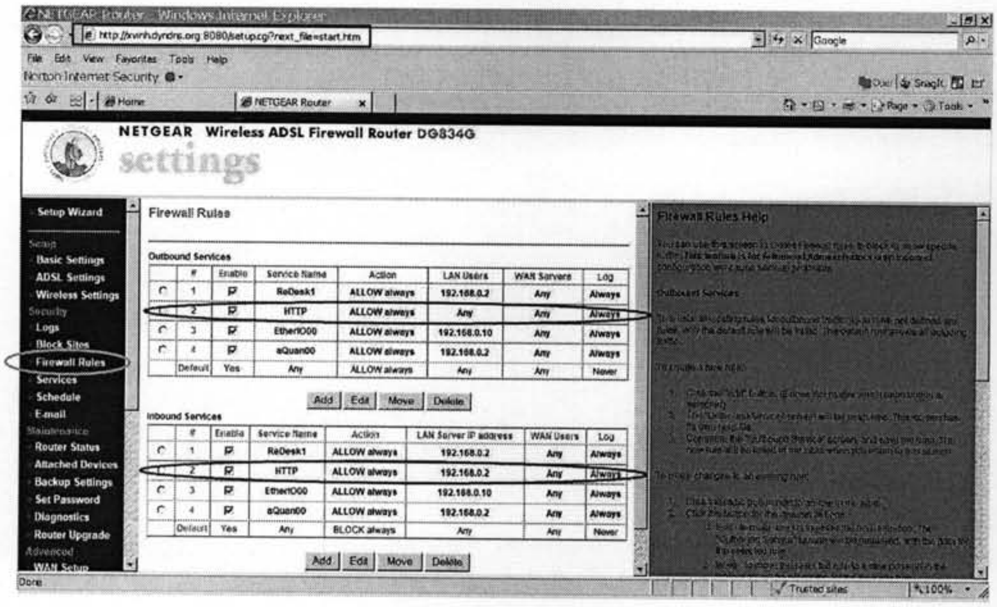
Figure 4.34 Event Log Page

The administrator can choose to display the event log of any system's user by selecting the user from the drop-down list. Intended date and time for displaying the log can also be manually picked. The wanted date and time must follow the format written within the brackets on the right.

In Figure 4.34, the event log is chosen to display for all users at all times. As mentioned in the database part, all the information related to one event is also displayed accordingly.

4. Security Management

o Router Level



NETGEAR Wireless ADSL Firewall Router DG834G

Firewall Rules

Outbound Services

	#	Enable	Service Name	Action	LAN Users	WAN Servers	Log
○	1	☑	ReDesk1	ALLOW always	192.168.0.2	Any	Always
○	2	☑	HTTP	ALLOW always	Any	Any	Always
○	3	☑	Ether1000	ALLOW always	192.168.0.10	Any	Always
○	4	☑	aQuan00	ALLOW always	192.168.0.2	Any	Always
	Default	Yes	Any	ALLOW always	Any	Any	Never

Inbound Services

	#	Enable	Service Name	Action	LAN Server IP address	WAN Users	Log
○	1	☑	ReDesk1	ALLOW always	192.168.0.2	Any	Always
○	2	☑	HTTP	ALLOW always	192.168.0.2	Any	Always
○	3	☑	Ether1000	ALLOW always	192.168.0.10	Any	Always
○	4	☑	aQuan00	ALLOW always	192.168.0.2	Any	Always
	Default	Yes	Any	BLOCK always	Any	Any	Never

Figure 4.35 Router Configuration Page

Router-level, or the Internet-level security management, ensures the secure connections to any computer located within the LAN network of the router itself by using the computers' IP addresses. This function is provided in a separate page from the main system page. When we go to the router page, then the Firewall Rules, the screen shown in Figure 4.35 will appear.

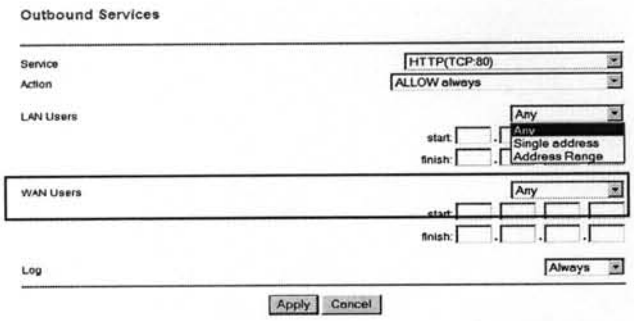
Connections to the server computer conform to the HTTP protocol and they are the HTTP requests. The router allows us to configure the HTTP requests from which IP on the Internet can receive the response from the server computer.

From the Inbound Services, as in Figure 4.35, WAN users are set to **Any**. This means any IP address on the Internet can send requests to and receive the responses from the server computer.

From the Outbound Services, as configured, any LAN computer behind the router can send out HTTP responses and any WAN computer (meaning that computers on the Internet) can receive the HTTP responses from the LAN computer.

For both inbound and outbound services, accesses can also be limited to only one IP address or to a range of IP addresses, depending on how the administrator wants to manage the system. Details are in the Figures 4.36 and 4.37.

- *For outbound services*

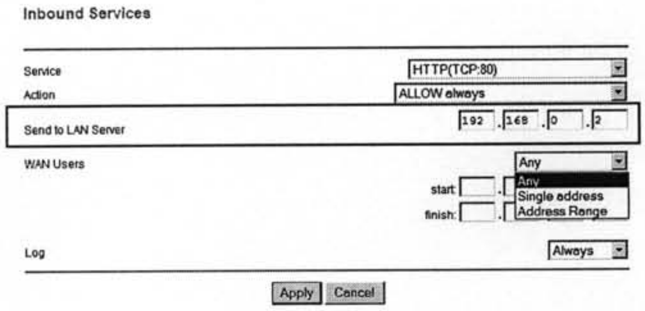


Outbound Services

Service: HTTP(TCP:80)
Action: ALLOW always
LAN Users: Any
start: . finish: .
WAN Users: Any
start: . . . finish: . . .
Log: Always

Figure 4.36 Outbound Services Settings

- *For inbound services*



Inbound Services

Service: HTTP(TCP:80)
Action: ALLOW always
Send to LAN Server: 192 . 168 . 0 . 2
WAN Users: Any
start: . finish: .
Log: Always

Figure 4.37 Inbound Services Settings

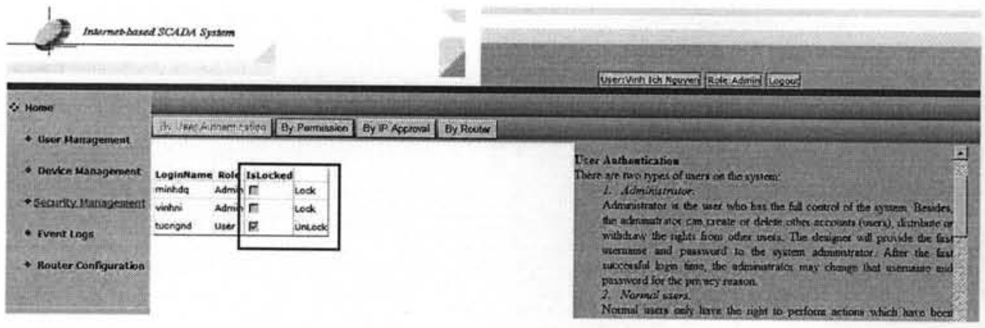
From these two figures, we see all the HTTP requests are directed to the address 192.168.0.2, which is the local IP address of the server computer, and every client computer on the Internet is allowed to send HTTP requests to the server computer. We can also configure the router to allow or block requests always or by schedule by using the Action field.

o **Server Application**

At this level, we are going to have three following tools:

- ***User Authentication***

Users with authenticated username and password can gain access to the system. However, the existing authenticated users might also be blocked by the administrator. By going to the Security Management/By User Authentication, we have Figure 4.38. To lock or release a user, the administrator needs to push the corresponding **Lock** or **Unlock** button. The status will then be displayed here as it is displayed on the User Manage management page showing whether a user has been locked or not.



LoginName	Role	IsLocked	
minhdq	Admin	☐	Lock
vinhni	Admin	☐	Lock
tuongnd	User	☑	UnLock

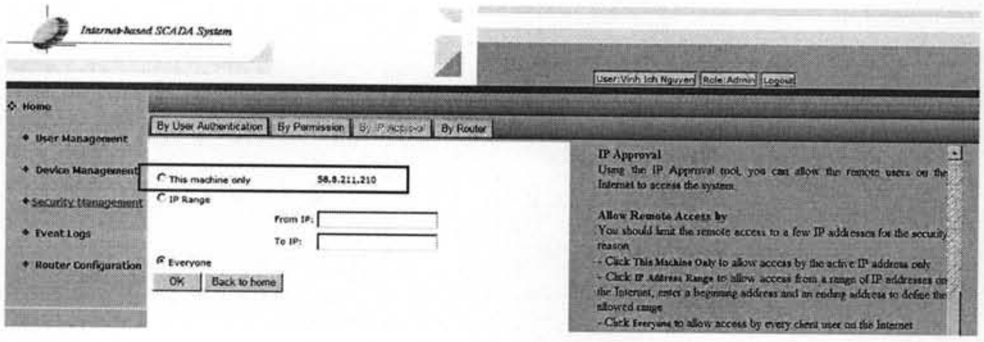
User Authentication
There are two types of users on the system:

1. *Administrator.*
Administrator is the user who has the full control of the system. Besides, the administrator can create or delete other accounts (users), distribute or withdraw the rights from other users. The designer will provide the first username and password to the system administrator. After the first successful login time, the administrator may change that username and password for the privacy reason.
2. *Normal users.*
Normal users only have the right to perform actions which have been

Figure 4.38 User Authentication Tool

- ***IP Approval***

There are three options for the system administrator to choose as shown in Fig 4.39.



This machine only 58.8.211.210
IP Range
From IP:
To IP:
[✓ Everyone]

IP Approval
Using the IP Approval tool, you can allow the remote users on the Internet to access the system.

Allow Remote Access by
You should limit the remote access to a few IP addresses for the security reason.
- Click This Machine Only to allow access by the active IP address only
- Click IP Address Range to allow access from a range of IP addresses on the Internet, enter a beginning address and an ending address to define the allowed range
- Click Everyone to allow access by every client user on the Internet

Figure 4.39 IP Approval Tool

- This machine only:

This is the IP address of the active client computer. If this option is chosen, no computers on the Internet can gain access to the system except only this specified machine. In the above example, we see the IP 58.8.211.210. If we are using the server computer itself to work on the system, this value is 127.0.0.1.

- IP Range:

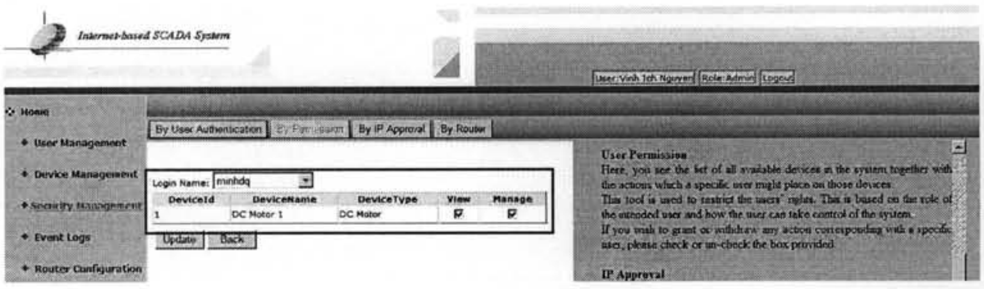
Only the computers whose IP addresses lie in the range specified in this option can gain access to the server.

- Everyone:

In this case, any client user on the Internet is allowed to connect to the system. However, this option is not suggested for a secure network.

- ***User Permission***

This tool is provided for the administrator to distribute specific rights on each device for each user. Details are shown in Figure 4.40.



Login Name: minhdq

DeviceId	DeviceName	DeviceType	View	Manage
1	DC Motor 1	DC Motor	✓	✓

User Permission

Here, you see the list of all available devices in the system together with the actions which a specific user might place on those devices.

This tool is used to restrict the users' rights. This is based on the role of the intended user and how the user can take control of the system.

If you wish to grant or withdraw any action corresponding with a specific user, please check or un-check the box provided.

IP Approval

Figure 4.40 User Permission Tool

With the simulated DC motor, two actions which are **View** and **Manage** are defined. **View** means that users can only view the devices' data, but not able to give any control command. **Manage** means that users can both view data and perform control actions on devices.

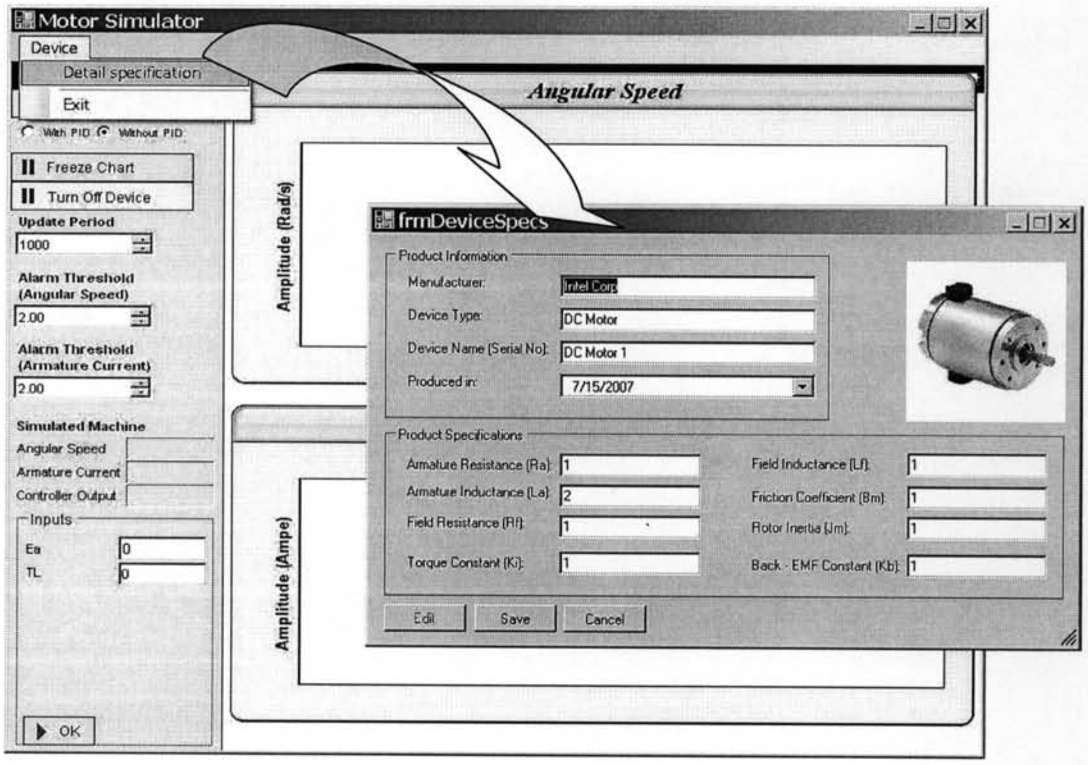
To work on this tool, the administrator needs to select a user and check the box(es) which correspond(s) to the action.

5. Device Simulator Operation

DS modules are written in Visual C # and run together with Matlab. Interface is also provided. Before working on the DS, users need to insert the specification information of the motor. The menu *Device* pops up the configuration window as shown in Figure 4.41, all the information in this window is required to be filled.

The product information and the specifications are to be understood for better operation. Each device has its own specification data which need to be recorded and maintained. However, in this research, we are working on the simulation scheme and thus the DS module is programmed to allow users to edit (by pressing the **Edit** button) this information and save (by pressing the **Save** button). Due to the reason that these specification data will be used in the Command Handler sub-module for calculating the response of the simulated DC motor, any change made in these data might cause unwanted reaction. Therefore, these data will be fixed by the system designer at one time and may never be changed during system operation.

Specification data about the devices must be stored somewhere. In the testing system, there will be a memory space in DS where these data are stored. However, for real applications with a huge number of devices, these data must be stored in the system's DB for better management.



With PID [✓ Without PID]

Update Period: 1000

Alarm Threshold (Angular Speed): 2.00

Alarm Threshold (Armature Current): 2.00

Simulated Machine

Angular Speed:

Armature Current:

Controller Output:

Inputs

Ea: 0

TL: 0

frmDeviceSpecs

Product Information

Manufacturer: Intel Corp

Device Type: DC Motor

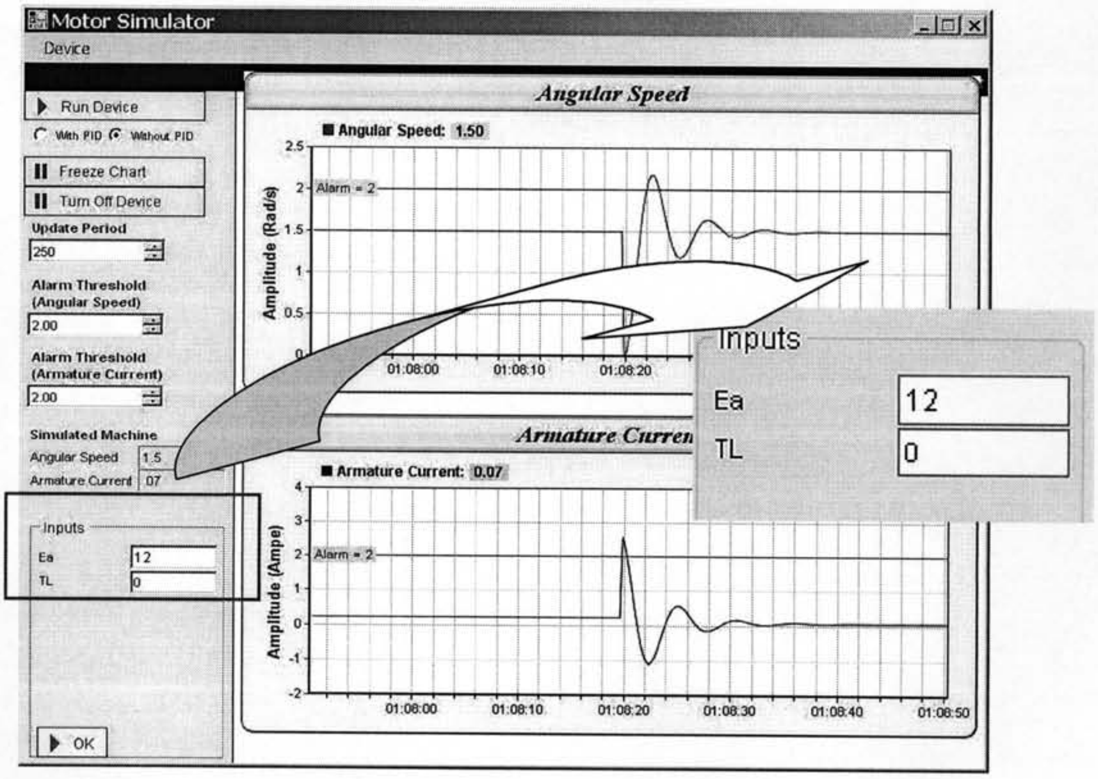
Device Name [Serial No]: DC Motor 1

Produced in: 7/15/2007

Product Specifications

Armature Resistance (Ra):	1	Field Inductance (Lf):	1
Armature Inductance (La):	2	Friction Coefficient (Bm):	1
Field Resistance (Rf):	1	Rotor Inertia (Jm):	1
Torque Constant (Ki):	1	Back - EMF Constant (Kb):	1

Figure 4.41 DS Specifications



With PID [✓ Without PID]

Update Period: 250

Alarm Threshold (Angular Speed): 2.00

Alarm Threshold (Armature Current): 2.00

Simulated Machine

Angular Speed: 1.5

Armature Current: .07

Inputs

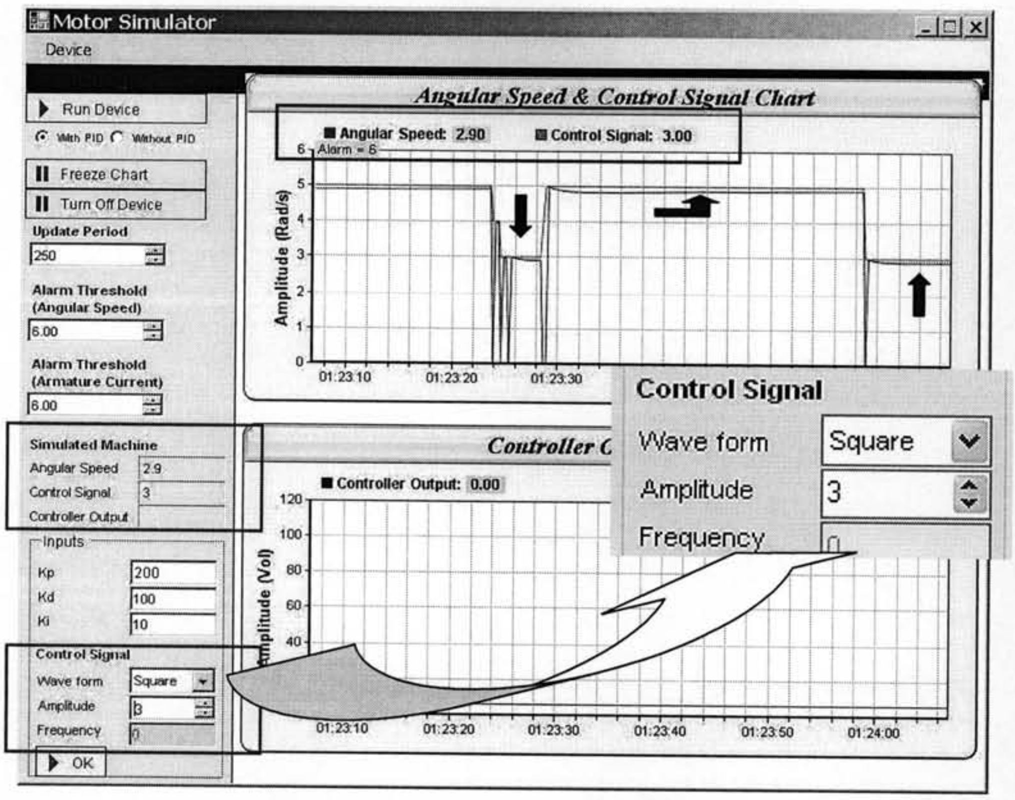
Ea: 12

TL: 0

Angular Speed: 1.50

Armature Current: 0.07

Figure 4.42 DS Operation under WithoutPID scheme



(● With PID ○ Without PID)

Update Period: 250

Alarm Threshold (Angular Speed): 6.00

Alarm Threshold (Armature Current): 6.00

Simulated Machine

Angular Speed: 2.9

Control Signal: 3

Controller Output:

Inputs

Kp: 200

Kd: 100

Ki: 10

Control Signal

Wave form: Square

Amplitude: 3

Frequency: 0

■ Angular Speed: 2.90 ■ Control Signal: 3.00

Alarm = 6

■ Controller Output: 0.00

Figure 4.43 DS Operation under WithPID scheme

- **WithouPID**

As shown in Figure 4.42, the DS is first opened under the WithoutPID scheme. Users need to specify the applied voltage and the load torque under which the motor should run. Pressing the button **Run Device** displays the data about the response of the simulater motor with the E_a and T_L values given

- **WithPID**

To move to the WithPID operation scheme, users need to turn off the simulated motor by pressing the **Turn Off Device** button.

Note that when this button is pressed, the Matlab m-file is stopped but the DS module interface is still open. When the motor is stop, users need to insert the values of K_p, K_i, K_d which are the parameters of the PID controller.

See Figure 4.43 for the operation of the DS module under WithPID scheme.

In this case, the simulated motor runs without load. In Figure 4.40, the speed of the motor has been regulated from 3 rad/s to 5 rad/s and then 3 rad/s again as desired by the active user. This is done by changing the amplitude of the control signal.

** On the DS interface, similarly to the SCADA pages, DS's data are displayed both graphically and numerically (See Figure 4.42 and Figure 4.43)*

** Looking back to Figure 4.31 and 4.32 for the data display on client SCADA pages, we can clearly see that the graphs of data here are not as smooth as the graphs on DS interface. This is because the delay time for each data point when traveling over the Internet from the DS module to the client SCADA pages is different from the others.*

This problem is common for every Internet-based system. This problem might be solved when the communication bandwidth at both server side and client side are increased. However, the data values are correct (see the numerical display) and we can still achieve a smooth data graph most of the times.

4.4 Cost Estimation

4.4.1. From the Developer Side

1. Hardware Components

There are 3 computers used in the experimental system.

- Server Computer with licensed Windows XP : 1 set *Price*: $600
- DS Computer with licensed Windows XP: 1 set *Price*: $400
- Client Computer with licensed Windows XP: 1 Set *Price*: $400

Total amount: **$1,400** **(1)**

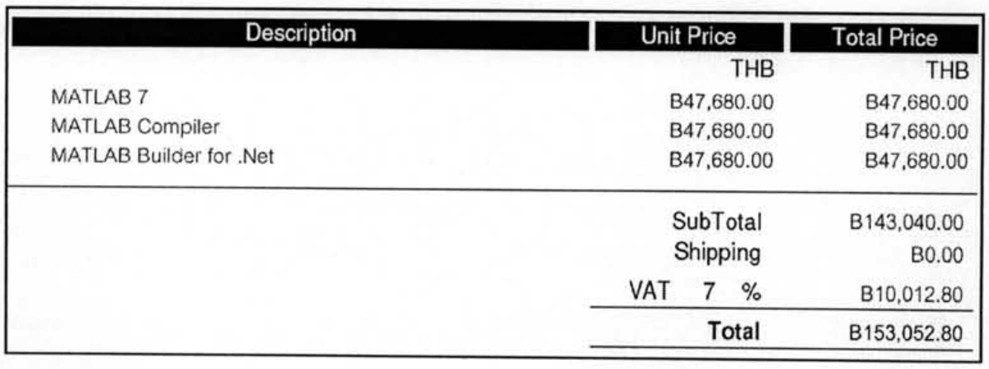
2. Software Components

Three software tools to be purchased: **Visual Studio 2005; Matlab** and **MySQL.**

- **Matlab**

The price of Matlab 7 and related products are quoted as in the Table 4.2.

Table 4.2 Matlab Product Price



Description	Unit Price	Total Price
	THB	THB
MATLAB 7	B47,680.00	B47,680.00
MATLAB Compiler	B47,680.00	B47,680.00
MATLAB Builder for .Net	B47,680.00	B47,680.00
	SubTotal	B143,040.00
	Shipping	B0.00
	VAT 7 %	B10,012.80
	Total	B153,052.80

* *This price is offered on April 19, 2007 by* TECHSOURCE, *the official distributor of Mathworks products in South East Asia.*

This research used only the Matlab 7 with the price as 47,680 THB, which is approximately equal to **$1,490**. If the COM technology is used, the developer has to consider the prices of the Matlab Compiler and Matlab Builder for .NET.

- **MySQL**

MySQL is provided by MySQL AB, a Swedish Company. Similar to .NET Products, there are many versions of MySQL products.
For reference, the Enterprise Edition is offered at $595/Server/Year.

In this research, the *free-of-charge and unlimited time* version was used.

- **Visual Studio 2005**

Table 4.3 lists the most common package offered by Microsoft on July 8, 2007.

Table 4.3 .NET Programming Products

Name	Illustration	Price
Visual Studio .NET Professional 2003 Full Retail Version		**$892.95**
Visual Studio Team Suite 2005 w/MSDN Premium Renewal 2 Yrs		**$4,995.95**
Visual Studio Pro w/MSDN Pro 2005 2 Yr Subscription		**$1,658.95**

In this research, the Visual Studio/MSDN Pro 2005 is used with price of **$1,658.95**

Total amount: $3,148.95 **(2)**

3. **Developer Payment**

Thesis Completion Period: 07 months

❖ Everyday work: 2 persons required

. SCADA Interface: 1 person

. Matlab Simulation and System Integration: 1 person

. Payment: $700/person/month * 2 (persons) * 7 (months) = **$9,800**

❖ Consulting Fee: (Referred from www.consultantjournal.com)

. Consultants: 1. Assoc. Prof. Dr. Watit Benjapolakul

2. Assoc. Prof. Krisada Visavateeranon

. Consulting Period: 1 hour/week

. Fee: $100/hour * 4 (weeks) * 7 (months) = **$2,800**

Total amount: **$12,600** **(3)**

Total Cost: (1) + (2) + (3) = $17,148.95 **(4)**

The number given in (4) is the estimated expense until the completion of the research. This amount of money is only approximated and it does not mean to be correct in a real system implementation. All the software tools can be used in the development of many other applications requiring .NET programming and Matlab simulation. Therefore, the given number is for reference only.

4.4.2. From the Customer Side

This research is to prove the ability of bringing the Internet into real applications. With a specific application, depending on the customers' real requirements plus the complexity of each application, we might need to add more hardware components and more efforts on writing the system's software. Therefore, the cost to be spent varies dependently on each application.