Supervisory Control and Data Acquisition System Design for CO2 Enhanced Oil Recovery

XIE LU

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2014-123
http://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-123.html

May 21, 2014

University of California, Berkeley College of Engineering

MASTER OF ENGINEERING - SPRING 2014

Department of Electrical Engineering and Computer Sciences

Supervisory Control and Data Acquisition System Design for CO_2 Enhanced Oil Recovery

Xie Lu

This Masters Project Paper fulfills the Master of Engineering degree requirement

Approved by:

1. Capstone Project Advisor:

Signature: ______________________________ Date: ____________

Print Name: Donald Wroblewski
Department: Fung Institute for Engineering Leadership

2. Faculty Committee Member #2:

Signature: ______________________________ Date: ____________

Print Name: Ruzena Bajcsy
Department: Electrical Engineering and Computer Sciences

Contents

List of Tables

List of Figures

Supervisory Control and Data Acquisition System Design for CO_2 Enhanced Oil Recovery

Xie Lu

Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA, USA
xielu@berkeley.edu

April 20, 2014

Abstract

Supervisory control and data acquisition (SCADA) are essential for oilfield operations. Currently available SCADA systems often do not incorporate recent technological advances that allow efficient network administration, over-the-air (OTA) programmability, and easy scalability to thousands of locations. Additionally, today's SCADA systems are not cost-effective for wells with low production rates such as CO_2 enhanced oil recovery (EOR). This paper proposes a SCADA system design that is based on a popular hybrid single-board computer. The controller is connected to a local wireless Ethernet/IP network with a secure IP-addressable radio and subsequently to the corporate intranet at the field central processing facility. The hybrid acts as a powerful and low-cost core for the controller. It is programmable in Python, and it is flexible, with provisions for digital outputs and most sensor inputs, as well as wireless sensor connections over Wi-Fi or Xbee radio standards. Sensor data are acquired and processed prior to being uploaded to a central database. To access the wellhead data, users log into a web-based graphical user interface (GUI). Similarly, the network structure allows users with proper permissions to send commands to, reprogram, and access the file system of individual controllers. Furthermore, the system applies an optimized control algorithm for sucker-rod pumps based on the Everitt-Jennings algorithm[1]. A prototype unit was deployed in a North Dakota oilfield with a total material cost of less than $1000. The system has applications in the oilfield beyond artificial lift control and wellhead sensing. With appropriate programming, the same hardware can be used for tank level monitoring, process condition monitoring, and alarming. The high level of control and programmability allows for efficient responses to dynamic oilfield conditions.

1 Introduction

1.1 Supervisory Control and Data Acquisition System

The supervisory control and data acquisition (SCADA) system is generally considered a necessary part of monitoring and control for large processes, including oil and gas production, paper manufacturing, and power generation. The system often consists of hardware architecture and a software package.

From the hardware perspective, the system mainly includes remote terminal units (RTUs) and master terminal units (MTUs). The core part of a RTU is a programmable logic controller (PLC), such as a micro-processor or single-board computer. RTUs are responsible for acquiring remote data from sensors or other data sources and implementing control strategies. MTUs are in charge of handling data processing and human-machine interaction[2]. The RTUs are connected to MTUs with SCADA system communication channels, as shown in Figure 1.

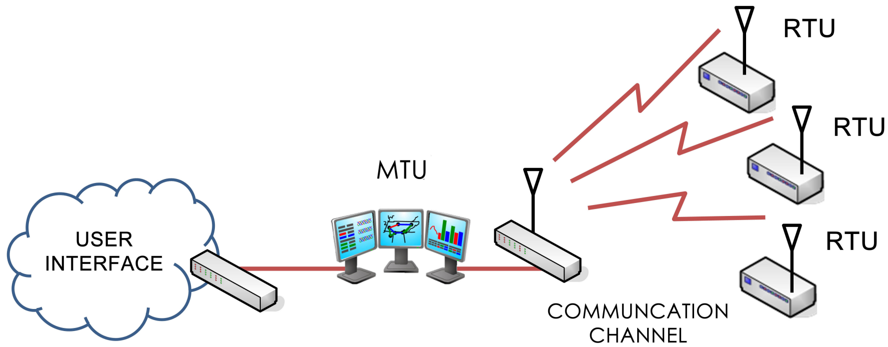

Figure 1: General SCADA System Architecture

The software package contains scripts running on PLCs, programs and databases on MTUs, as well as graphical user interface (GUI) for human-machine interaction. The software package should be designed to satisfy all kinds of customized needs and to achieve predefined functionalities. However, with the popularity of World Wide Web (WWW), there is an increasing need for SCADA systems to incorporate web-based GUIs for access via Internet[3].

1.2 CO_2 Enhanced Oil Recovery

Typical ways of producing oil include propelling with natural pressure and water flooding. Nevertheless, conventional methods will only procure about 30% of the initial oil from a reservoir, leaving at least 50% of the initial resource still in the ground[4]. The volatile price of oil and the desire to decrease foreign dependency have created more attention around this kind of oil we currently have stored underground. Enhanced Oil Recovery (EOR) is an unconventional way to extract additional oil from mature oilfields. Figure 2 is a diagram of CO_2 EOR process.

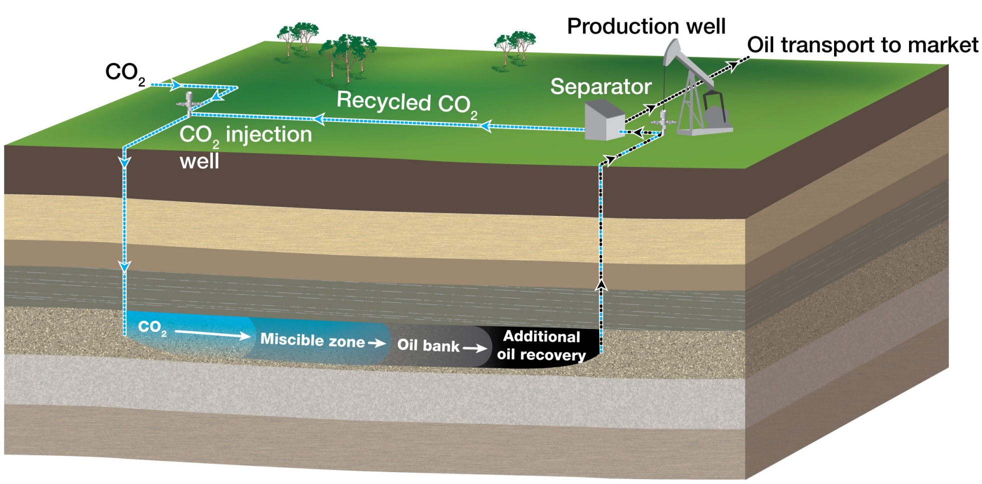

Figure 2: CO_2 EOR Diagram[1]

[1]Image source: Australian Government's Cooperative Research Centre's Program; CRC for Greenhouse Gas Technologies. Available at: `http://www.co2crc.com.au/images/imagelibrary/stor_diag/EOR_media.jpg` [Accessed 12 April 14].

CO_2 is a good solvent for oil recovery because it is miscible with oil and can be obtained at a low cost. By injecting CO_2 into oil reservoirs, near carbon-neutral oil will be produced. CO_2 EOR has been used for the past two decades and is one of the most environmentally sound methods that can be used to extract additional oil[5]. At the end, the wells are capped and the CO_2 remains sequestered in the stable geological formation that previously held oil and natural gas for millions of years.

Like traditional ways of oil production, CO_2 EOR requires a SCADA system to monitor and control the process. Unfortunately, the production rate of CO_2 EOR is relatively low, and thus currently available SCADA systems are not cost-effective for hundreds of thousands of wells with low production rates. In order to convert these dormant wells into producers, the design of a customized and cost-effective SCADA system for CO_2 EOR is expected.

1.3 System Requirement

The SCADA system design for CO_2 EOR proposed in this paper is anticipated to meet the functional requirements from C12 Energy, an oil and gas company which is rehabilitating mature fields through gas injection. The devices need to be designed to meet the specific needs of the company with flexibility for use in future systems. To obtain economic feasibility for use in oilfield operations, the design needs to be scalable to hundreds of thousands of wellheads at a unit price below $1,000 after installation. While the system is designed for use by C12 Energy, we are anticipating to develop the final product for market. Determined sensors will be incorporated in our design but the system is created with flexibility to be incorporated into future SCADA systems.

For easy set-up and updates, it is important that the system is capable of being programmed over the network. Configuration of radio devices can be done before deploying them to the field. Once in the field the devices must operate at a high level of reliability. Local monitoring and recording are ideal as a backup for system failure. The network used in creation of this system must be designed with security and reliability in mind. The user needs to be able to access the devices at any time. Unauthorized access could result in external control of oil wells. The user would best utilize the system if it were programmable in common languages such as Python. The device also must be capable of accepting manual control and inputs. In addition, a MySQL database is preferred for establishing efficient data storage structure and implementing human-machine interaction.

2 Literature Review

2.1 Remote Terminal Unit

RTUs are an essential part for achieving distribution automation in SCADA systems. They collect in-field data from analog or digital sensors and other data sources at each remote site and send data back to MTUs via a communications system[6]. They are also in charge of outputting analog or digital signals to actuators to realize feedback control.

However, traditional RTUs often do not have the ability to do calculations and only support a limited number of communication protocols. They are like human beings who have eyes, ears and hands, but no heart[7]. Therefore, data storage, processing, and calculations have to be done on MTUs. This brings a lot of potential risks and issues such as control dropout when communication between RTUs and MTUs fails and the efficiency of feedback control is constrained by network speed, load, and throughput.

Following the rise of PLCs such as micro-processors or single-board computers, RTUs are more likely to include PLCs to reduce overall product cost, improve flexibility and enhance performance, because it is possible to off-load the communication channel and the MTUs by performing calculations locally[7].

Early PLCs are normally programmed in a visual language called ladder logic[8], whereas recent microprocessors and single-board computers support advanced programming languages such as C, C++, and Python. These newly-developed instances of RTUs are essentially PC systems with CPUs, random access memory, nonvolatile memory, I/O ports, analog-to-digital converters, and network infrastructure, minimizing the learning curve for engineers who are already experienced with desktop PCs[9].

2.2 Master Terminal Unit

MTUs, also known as data servers, host computers, or base stations, are an indispensable part of modern SCADA systems. They connect through communication protocols to RTUs for data and information exchange[10]. Modern MTUs are essentially reliable industry computers which gather data from RTUs, store data, and host service for human-machine interaction. In certain cases, MTUs also need to process data and give feedback control signals to RTUs.

Recent SCADA systems often have both database servers and application servers for MTUs[11]. RTUs are regarded as clients and provide information and resources required by servers[12]. Database servers typically contain a relational database management system (DBMS) such as MySQL, Oracle, SQL-Server, or Access that is used for storing a huge amount of field data and supporting a variety of user queries[13]. With hundreds of thousands of sensors generating data continuously, cloud storage and cloud computing might be introduced to future SCADA systems to provide solutions in the cloud environment[14]. Application servers are required to provide human-machine interaction services such as data monitoring and manual control. Both servers have to deal with access control management to serialize access to shared resources by multiple clients[15].

2.3 Communication Channel

RTUs and MTUs are interconnected through a variety of communication channels, which will constrain the speed and security at which data acquisition and control can be performed[16]. These channels typically include radio links, leased lines, phone lines, fiber optics, cables, and/or a combination of these techniques[17]. Special attention should be paid to the revolution of WWW since the late 1990s[18]. Ethernet TCP/IP, the backbone of WWW, has been preferred by modern SCADA systems to overcome the limitations of analogue communications and to be more flexible in terms of expansion and reconfiguration[19].

Emerging wireless technologies and deep research in wireless sensor network (WSN) make it possible to build wireless SCADA systems that provide efficient, convenient, and reliable distributed information processing and long-distance monitoring and control[20]. Wireless network is becoming increasingly widely used in industry for data and information exchange. Technologies include GSM, GPRS, 3G/4G, UMTS, WiMax, MBWA, Wi-Fi, Bluetooth, ZigBee, and UWB[21]. Especially, the move to wireless Ethernet on the basis of IEEE 802.11 standard seems to be the next generation of WWW revolution in industrial communication networks[22], because it reduces the huge cost of wiring and provides a high degree of convenience and mobility, wired Ethernet does.

Other considerations in SCADA system communication channels include reliability, security, and flexibility. Several security issues have been brought to attention, including access control, firewalls, intrusion detection, cryptography, key management, as well as device and operating system security[23]. To increase flexibility and redundancy, communication should not be limited at the RTU-MTU level. Instead, data sharing and forwarding are expected to happen between different RTUs[24].

2.4 Sucker-Rod Pump Control

Sucker-rod pumping is one of the most widely used means of producing oil in the industry. Approximately two-thirds of the world's oil production wells are using sucker-rod pump systems[25]. Sucker-rod pump systems are usually composed of three parts including a pumping unit, sucker rod, and pump[26]. Undoubtedly, many methods have been proposed to analyze and control these systems. One of the most significant issues in sucker-rod pump control is to shut off the motor for a given time period when a well has been determined pumped off, meaning that the fluid level has been at the pump, as shown in Figure 3.

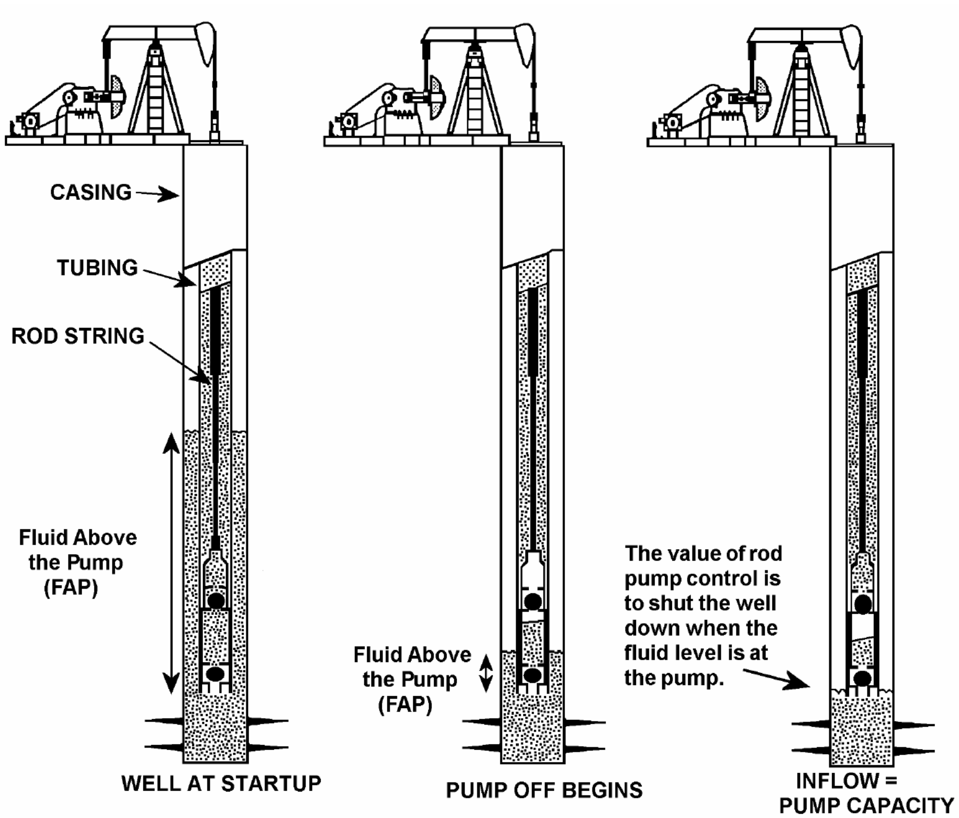

Figure 3: Sucker-Rod Pump Control Diagram[2]

The use of a surface dynamometer to monitor the power input to the rod string has been the primary method of monitoring a rod pumped well and determining when the well has pumped off in sucker-rod pumping systems[27]. This enables computer calculation and diagnosis of down-hole conditions such as intermediate rod stresses, pump intake pressure, unanchored tubing, excessive rod friction, leaking pumps, and production strings[28]. Surface dynamometers also make it possible to automate feedback control in sucker-rod pumping wells[28].

However, this method is not accurate enough because the ideal analysis procedure uses a down-hole dynamometer instead a surface one[1]. Unfortunately it is not cost-effective to measure down-hole data because this often requires installation of equipment thousands of feet below ground. Therefore, a method of computing down-hole dynamometer cards using finite differences based on surface data has been proposed[29]. Everitt and Jennings proposed a new approach on the basis of finite-difference representation of the wave

[2]Image source: Available at: `http://www.ep-weatherford.com/PDF/Papers/Value_of_Rod_Pumped_Control.pdf` [Accessed 12 April 14].

equation and a consistent method of computing the viscous damping term associated with the damped-wave equation[1].

Advanced analytical approaches and algorithms, including expert systems, rough set theory, artificial neural networks, supported vector machines, spectrum analysis, and filter techniques, have been proposed recently to diagnose down-hole conditions[30]. Other analytical methods are aimed to detect multiple down-hole faults that may happen at the same time[30].

Recent development of low cost downhole sensors has made it possible to measure downhole variables that will assist the sucker-rod pump control[31]. New feedback control strategies that are based on downhole parameters have been proposed to improve oil production and minimize operational costs[32]. The variable-frequency drive is used to "vary the speed of the electric motor driving the pump by detection of decreasing fluid in the pump", and thus "the electric motor is never turned off but varies in operating speed"[33].

2.5 Graphical User Interface

The revolution of WWW not only brings Ethernet TCP/IP into SCADA system communication channels, but also attracts more attention to improve the flexibility of traditional GUI design for better human-machine interaction experience. Conventional GUIs of various complexities have been developed for SCADA systems but they are mostly stand-alone programs running on independent machines[3].

A web-based GUI enables users to use their web browsers to monitor and control a system running inside a corporate intranet when they are outside the corporate network[34]. Such SCADA systems often require intelligent RTUs that are implemented using modern intelligent electronic devices and support HTTP protocol and MTUs with a web server/browser structure[11]. This kind of information layer is compatible with different kinds of operating systems, improves flexibility and reliability, and eliminates software, hardware and administrative costs at the same time[35].

The two most widely-used HTTP server platforms are Microsoft's Internet Information Service (IIS) and open-source Apache Server, both introduced in 1995[36]. Scripting languages include ASP.NET (C#. VB.NET), PHP, Perl, and a typical DBMS contains MySQL, Oracle, SQL-Server, and Access[37]. Recent development of NoSQL databases such as MongoDB and DynamoDB has improved the scalability, performance and availability of data storage and management and allowed cloud and desktop computing[38]. Security issues such as SQL injection and cross-site scripting must be taken into account when designing web-based SCADA system GUIs[39].

3 Methodology

3.1 System Overview

The architecture of our SCADA system for CO_2 EOR is displayed in Figure 4. Like most SCADA systems, the design consists of RTUs, MTUs, and communication channels. Each RTU is composed of a 120V power supply, a single-board computer called Arduino Yun, sensors and actuators positioned at each wellhead. MTU is essentially a central server which runs a MySQL DBMS and provides web service with Apache Server and PHP scripts in our model. RTUs and MTU are interconnected by wireless Ethernet/IP connections provided by Ubiquiti NanoStation. The data and information flow in the system is shown in Figure 5.

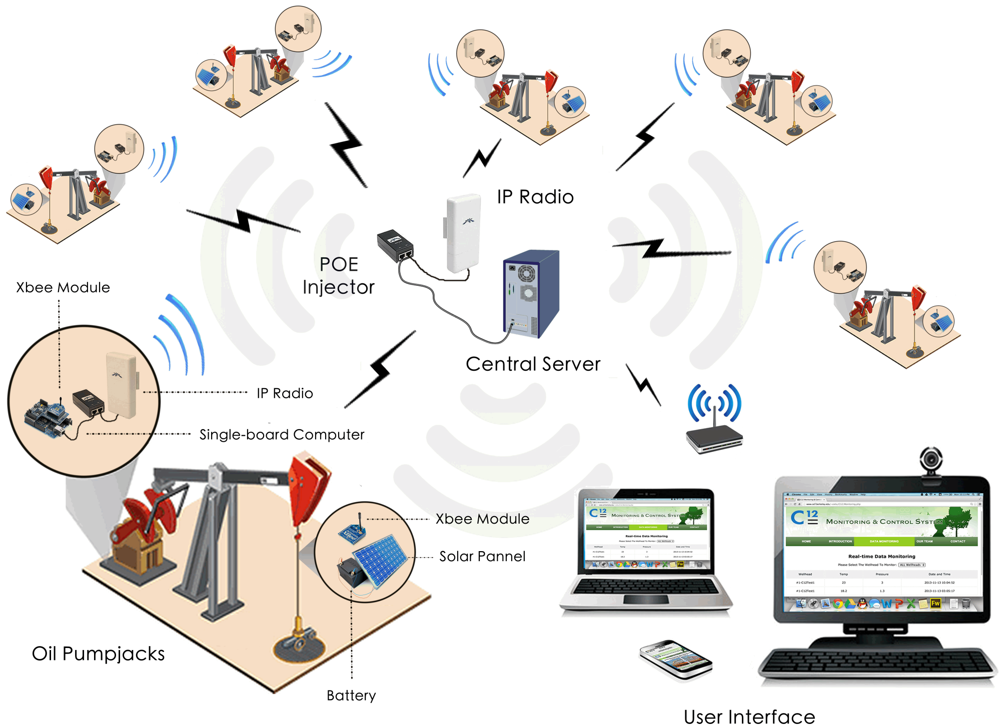

Figure 4: Architecture of the SCADA System for CO_2 EOR

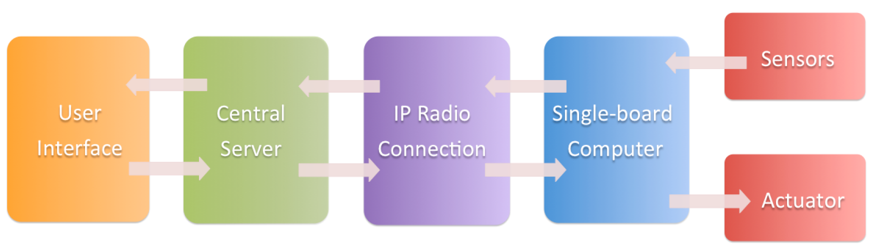

Figure 5: Data and Information Flow in the SCADA System for CO_2 EOR

Data acquisition will occur through the Arduino Yun, which will be connected to the sensors either with or without wires. Sensor data include tilt, temperature, pressure, current, acceleration, and force. A sucker-rod pump control algorithm is implemented on the controller, which will output signals to shut off the motor for a given time period when a wellhead has been determined pumped off.

The IP-addressable radios are physically connected to Arduino Yuns and the central server using their Ethernet ports. The radios will build a wireless local are network (WLAN) for data and information exchange between Arduino Yuns to the central server.

Wellhead data are stored in the MySQL database on the central server as well as SD cards of local single-board computers at the same time. Web service running on the central server will provide a GUI for human-machine interaction, including querying wellhead data, accessing Arduino Yun's operating system, updating control algorithm, and sending manual control command.

3.2 Remote Terminal Unit Design

The core part of our RTU design is a single-board computer named Arduino Yun, which is based the ATmega32u4 and the Atheros AR9331, as shown in Figure 6[40]. The Arduino Yun is a sensor node itself, with tilt, temperature, pressure, and current sensors connected directly to its I/O ports; it is also a data aggregator, collecting data from rod force sensor and accelerometer via a ZigBee wireless network using Xbee modules[41].

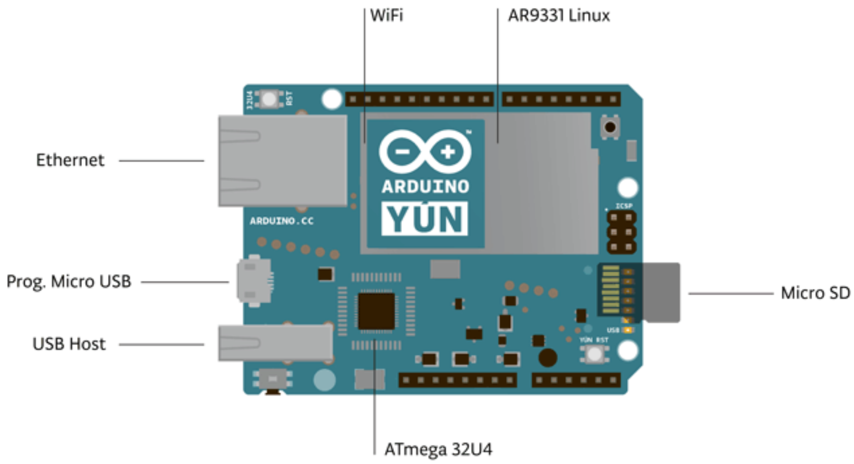

Figure 6: Arduino Yun Board[3]

What makes Arduino Yun stand out is that "the Atheros processor supports a Linux distribution based on OpenWRT named Linino"[40]. It offers a powerful networked computer with built-in Ethernet and WiFi support, a USB-A port, and micro-SD card slot[40]. The two processors can communicate with each other, as shown in Figure 7, and thus users can use access the Linux distribution through ssh and use Linux commands and write shell and python scripts for robust interactions[40].

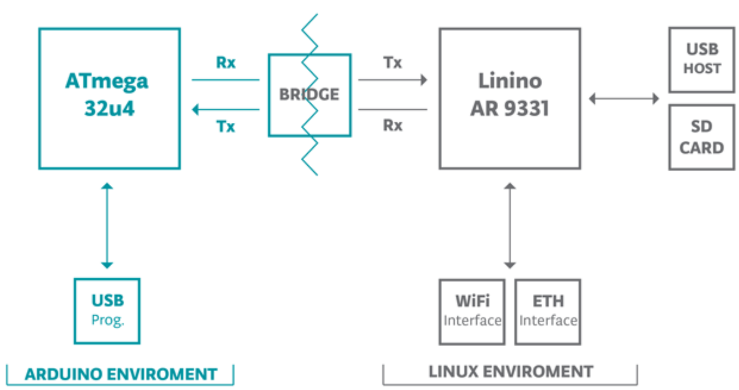

Figure 7: Arduino Yun Two Processors Communication[4]

[3]Image source: Available at: `http://arduino.cc/en/uploads/Main/YunParts.png` [Accessed 13 April 14].
[4]Image source: Available at: `http://arduino.cc/en/uploads/Main/BridgeInShort.png` [Accessed 13 April 14].

The main part of the RTU design is displayed in Figure 8. A prototype has been built with a NEMA 4 enclosure for the protection of the equipment against ingress of foreign objects like dirt, dust, rain, sleet, snow, and ice. The prototype unit has been deployed in a North Dakota oilfield with a total material cost of less than $1,000.

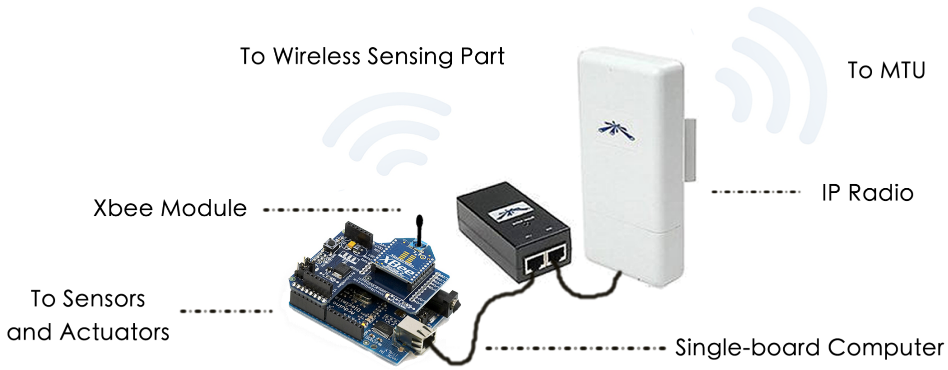

Figure 8: Main Part of the RTU Design

3.3 Wireless Sensing Part Design

The load cell sensor and the accelerometer are supposed to be wirelessly connected, as shown in Figure 9, because it is economically infeasible to connect them to the Arduino Yun via wires. The distance between these sensors and the Arduino Yun is around 50 meters so a ZigBee wireless network with Xbee modules is appropriate for data transmission.

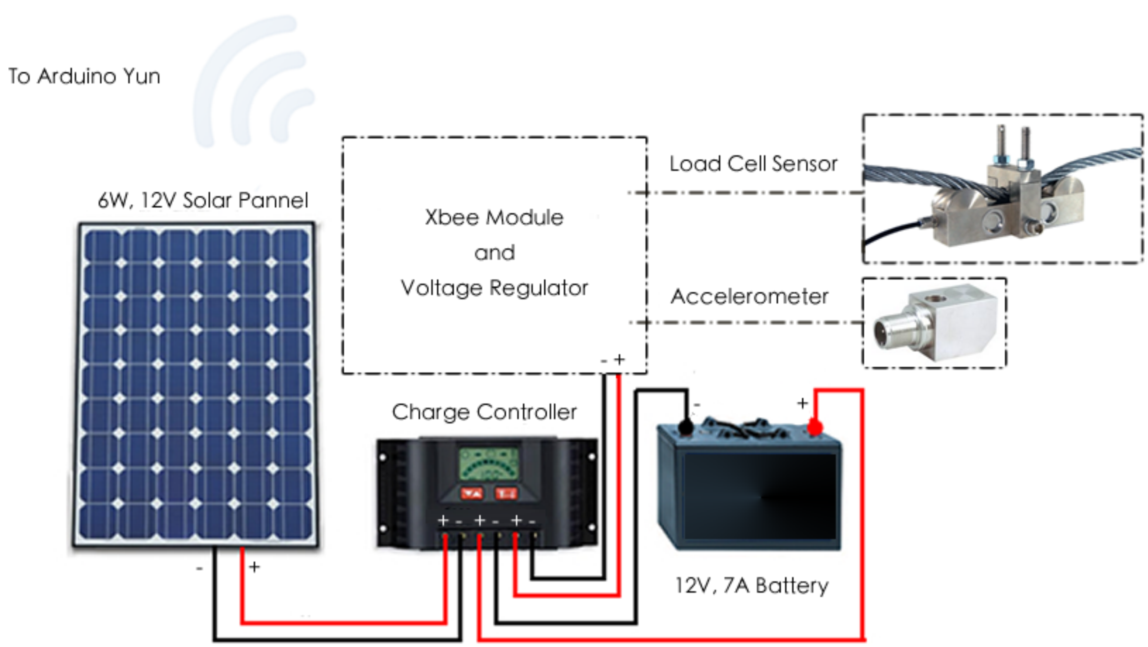

Figure 9: Wireless Sensing Part Design

To avoid frequently changing batteries in the oilfield, a solar panel is introduced. All the components are selected to lower the power consumption. We assume that there are 4 hours of sunshine per day on average and that the solar panel operates at 50% of the continuous power capability specified on the data sheet; we also assume that the Xbee operates 30% in transmit mode, 30% in receive mode, and 40% in sleep mode. Theoretically, under these assumptions the wireless sensing part will not need a battery replacement.

3.4 Master Terminal Unit Design

The MTU in our SCADA system is essentially a central computer with a Linux operating system. It is acting as a both database server and an application server. The computer is running a MySQL DBMS and providing web service with Apache Server and PHP scripts.

The database has four relations, or tables, including User, Wellhead, Data, and Log. The User relation stores user information of the SCADA system such as user ID, user name, password, permission, login time, and login IP. This relation is mainly used for authorization. The Wellhead relation keeps track of the information of hundreds of thousands wellheads, including wellhead ID, description, longitude, latitude, and status. The Data relation is for storage of all the wellhead data. This relation has many fields defined by the user, including tilt, temperature, pressure, current, acceleration, and force data. Each tuple or record has its unique record ID and its wellhead ID indicating to which wellhead the record belongs. Each record also has a date and time filed to indicate when it is added. The Log relation stores all kinds of system logs such as user operations so that it will be possible to investigate problems when system crashes. The structures of the four relations are shown in Table 1, 2, 3, and 4.

Table 1: The User Relation

Field Name	usrID	usrName	password	permission	loginTime	loginIP
Data Type	int(11)	varchar(20)	varchar(40)	int(1)	datetime	varchar(20)

Table 2: The Wellhead Relation

Field Name	wellID	wellDescription	longitude	latitude	status
Data Type	int(11)	varchar(50)	double	double	int(1)

Table 3: The Data Relation

Field Name	recordID	wellID	tilt	temp	pressure	current	acc	force	addTime
Data Type	int(11)	int(11)	double	double	double	double	double	double	datetime

Table 4: The Log Relation

Field Name	logID	usrID	operation	logTime
Data Type	int(11)	int(11)	text	datetime

PHP scripts running on the MTU have two primary functionalities. Firstly, they provide an interface for RTUs to upload data into the MySQL database. RTUs, or Arduino Yuns, can access this interface with parameters through Ethernet/IP connection in the WLAN. This kind of access is completed by a GET request with query strings in the URL like "`http://192.168.1.12/uploaddata.php?key=3A4K9E2M3D2&wellId=18&tilt=32.8&temp=21.3&pressure=1.3&current=2.6&acc=10.2&force=15.6`". Secondly, they provide GUIs for human-machine interaction. The results generated by these PHP scripts are normally in HTML, CSS, and JavaScript format, which can then be interpreted by users' browsers to display as GUIs.

3.5 Communication Channel Design

There are mainly three kinds of communication within our SCADA system for CO_2 EOR. The first communication happens within each RTU. The rod force sensor and the accelerometer sensor are connected to the Arduino Yun via a ZigBee wireless network with Xbee modules due to the lack of economically feasibility of using wires between the main RTU box and the sucker-rod.

Another interconnection occurs between RTUs and MTU via wireless Ethernet/IP connection provided by Ubiquiti NanoStation. Each Arduino Yun or central server is equipped with an Ubiquiti NanoStation connected to its Ethernet port. Ubiquiti NanoStation is an IEEE 802.11 solution, which "packs some phenomenal performance with a revolutionary design combining a hi-gain 4 antenna system, advanced radio architecture, and highly researched and developed firmware technology allowing throughput, stability, and capacity performance rivaling even the highest-end WiMax networks"[42]. This IP-addressable radio allows for a high wireless Ethernet/IP data transfer rate within more than 10 miles radius depending on the model type. Meanwhile, communication is not limited the RTU-MTU level because RTUs can talk to each other, which increases the flexibility and redundancy of the whole SCADA system.

The third communication happens when users utilize GUIs to interact with the SCADA system. Obviously the RTUs and MTU are forming a local area network (LAN), which is a corporate intranet. If the user's device is inside the same corporate intranet, he or she can access the GUI directly. Otherwise, the user has to establish a virtual point-to-point (P2P) connection with virtual private network (VPN). In this way, operators and engineers can monitor and control the system when they are outside the corporate network.

3.6 Sucker-Rod Pump Control Design

Sucker-rod pump control aims to shut off the motor for a given time period when a well has been determined pumped off based on down-hole condition auto diagnosis. In our system, this time period is set to be half an hour. Most auto diagnosis algorithms rely on down-hole data, which are not cost-effective to measure because this often requires installation of equipment thousands of feet below ground. Therefore, the Arduino Yun gathers data from up-hole rod force sensor and accelerometer via ZigBee network and calculates down-hole force and displacement based on the algorithm.

An optimized control algorithm for sucker-rod pump control based on the Everitt-Jennings algorithm[1] is implemented to deal with sucker-rod pumps that have multiple diameters at different positions. For sucker-rod pumps that have three diameters, assume that $A_k(k = 1, 2, 3)$ is rod cross-sectional area (in^2) for each diameter. Based on the Everitt-Jennings algorithm, the 1D wave equation accounting for variable rod diameters is given by:

$$E{A_k}^2 \frac{\partial^2 u}{\partial x^2} = \frac{\rho A_k}{144 g_c} \frac{\partial^2 u}{\partial t^2} + c \frac{\rho A_k}{144 g_c} \frac{\partial u}{\partial t} \quad (1)$$

Where u is the rod displacement (ft), x is the axial distance along the rod string (ft), t is time (sec), E is Young's modulus of elasticity (psi), g_c is units conversion factor ($32.2(lbm - ft)/(lbf - sec^2)$), ρ is rod density (lbm/ft^3), and c is damping coefficient (sec^{-1}).

Assume that i is the number of iteration, j is the number of time steps, k_1 and k_2 are iteration boundaries of the three diameters, and m is the last iteration where the point is just above the pump. According to the Everitt-Jennings algorithm, by substituting Taylor series approximations into the above equation, we have:

$$u_{i+1,j} = \begin{cases} \{[\alpha_1(1+c\Delta t)]u_{i,j+1} - [\alpha_1(2+c\Delta t) - \beta_1^+ - \beta_1^-]u_{i,j} + \alpha_1 u_{i,j-1} - \beta_1^- u_{i-1,j}\}/\beta_1^+, & 1 <= i < k_1 \\ \{[\alpha_2(1+c\Delta t)]u_{i,j+1} - [\alpha_2(2+c\Delta t) - \beta_2^+ - \beta_2^-]u_{i,j} + \alpha_2 u_{i,j-1} - \beta_2^- u_{i-1,j}\}/\beta_2^+, & k_1 <= i < k_2 \\ \{[\alpha_3(1+c\Delta t)]u_{i,j+1} - [\alpha_3(2+c\Delta t) - \beta_3^+ - \beta_3^-]u_{i,j} + \alpha_3 u_{i,j-1} - \beta_3^- u_{i-1,j}\}/\beta_3^+, & k_2 <= i <= m \end{cases} \tag{2}$$

Where

$$\alpha_k = \frac{\overline{\Delta x}}{\Delta t^2}\left[\frac{(\rho A_k/144g_c)^+ + (\rho A_k/144g_c)^-}{2}\right] \tag{3}$$

$$\beta_k = \frac{EA_k}{\Delta x} \tag{4}$$

This becomes a dynamic programing problem, where $u_{0,j}$ and $u_{1,j}$ are base cases that must be known for all time steps j. $u_{0,j}$ can be known from the surface dynamometer card, whereas $u_{1,j}$ can be inferred from Hooke's law. Detailed derivations can be found in Everitt and Jennings' paper[1]. The pump displacement $u_{pump,j}$ is finally given by:

$$u_{pump,j} = (1 + c\Delta t)u_{m-1,j+1} - c\Delta t u_{m-1,j} + u_{m-1,j-1} - u_{m-2,j} \tag{5}$$

Therefore, the pump load $F_{pump,j}$ can be calculated by:

$$F_{pump,j} = \frac{EA_3}{2\Delta x}(3u_{m,j} - 4u_{m-1,j} + u_{m-2,j}) \tag{6}$$

In this way, the curve of down-hole pump force versus displacement of a cycle is generated. The generated curve will be compared to the normal curve that has been preset into the system. If the generated curve goes beyond 10% of the preset normal area, the pump has pumped off, and the Arduino Yun will output proper signals to shut off the motor and restart it after thirty minutes when underground fluid reaches the pump capacity again.

3.7 Graphical User Interface Design

The web-based GUI is hosted by the Apache Server running on the MTU, as shown in Figure 10. It is programmed in PHP scripts based on CodeIgniter, a proven, agile and open PHP web application framework using Model-View-Controller (MVC) software pattern. To improve user experience, it fully utilizes recent web technologies such as HTML5, CSS3, and JavaScript, and incorporates most popular front-end framework such as Bootstrap.

To interact with the MySQL database on the MTU, a connection to the MySQL server must be opened with the *mysqli_connect*() function or using the *Database* class provided by Codelgniter. Access control is accomplished by querying the User relation and handling *session* to prevent access from unauthorized users or authorized users without required permission. Untrusted data and input parameters are being strictly validated before further processing to deal with common security issues in web applications such as cross-site scripting and SQL injection. The MD5 message-digest algorithm has been utilized to better protect user passwords. Asynchronous Java Script and XML (AJAX) technology is utilized to create an asynchronous GUI where the browser can send data to and retrieve data from the MTU asynchronously without interfering with the display and behavior of the existing page[43].

Figure 10: Graphical User Interface Design

Other GUIs include the Arduino IDE, which can be used as an alternative for updating programs running on each Arduino Yun. Ubiquiti also provides a web interface for NanoStation to configure the network. Experienced engineers can use Secure Shell (SSH) directly to access each Arduino Yun for remote command-line login and command execution[44].

4 Discussion

4.1 Unit Testing

We have conducted unit testing on the hardware components we chose, including Arduino Yun, Ubiquiti NanoStation, and Xbee module, as well as our software package that handles data and information processing. Unit testing is mostly based on the data sheet or user manual provided by the vendors. It is aimed to validate the functionality and reliability of these individual components or modules before they are assembled together.

The single-board computer unit testing includes checking, for example, the digital and analog input and output, the power supply, the bridge communication between two processors, and the OTA programmability. The Ubiquiti NanoStation testing contains the validation of radio communication ability, network range, speed, load, and throughput. The Xbee module testing includes checking both digital and analog input and its connection to the single-board computer. The software package testing is performed by simulating the regular operational environment as well as some extreme cases, such as SQL injection and cross-site scripting, to test its security and reliability.

The unit testing results indicate that these individual parts function properly and satisfy our expected functional requirements. Unit testing is necessary before moving forward, because once these components are assembled, it is not easy to investigate what exactly causes a problem.

4.2 System Testing

Another significant step before building the prototype was to conduct system testing to ensure that individual components will work properly when they are assembled together. A simplified system was built to perform the system testing. It is composed of all the core parts of our SCADA system, including a temperature sensor, a tilt sensor, a single-board computer, Arduino Yun, three IP-addressable radios, Ubiquiti NanoStation, two Xbee modules, an MTU server with our software package installed, and a client PC.

Data acquisition occurs through the Arduino Yun, which is connected to the temperature sensor wirelessly using Xbee modules and connected to the tilt sensor with wires. Sensor data are gathered and stored in the MySQL database on the MTU server as well as the SD card of the Arduino Yun. The three IP-addressable radios are physically connected to the Arduino Yun, the MTU server, and the client PC respectively using their Ethernet ports. Web service running on the MTU server provides a web-based GUI for human-machine interaction. Authorized users are able to monitor and control the whole system using the client PC, which is in the system's LAN.

The system testing results demonstrate that the assembled system behaves as expected. Nevertheless, attention should be paid to the two major differences between our testing condition and the reality. Firstly, The number of wellheads is much higher in reality because the system will be scaled to hundreds of wellheads in the field. This significant increase in load will probably change the network performance. Additionally, it is likely that topographic features and hostile environments in the field will affect the stability of the system. For example, hills and mountains may weaken radio signals, and low or high temperatures may cause hardware abnormality.

4.3 Prototype Testing

After conducting the system testing, a prototype of our SCADA system for CO_2 EOR has been built and deployed in C12 Energy's North Dakota oilfield. Figure 11 displays the RTU of the SCADA system prototype, and Figure 12 shows the wireless sensing part of the SCADA system prototype.

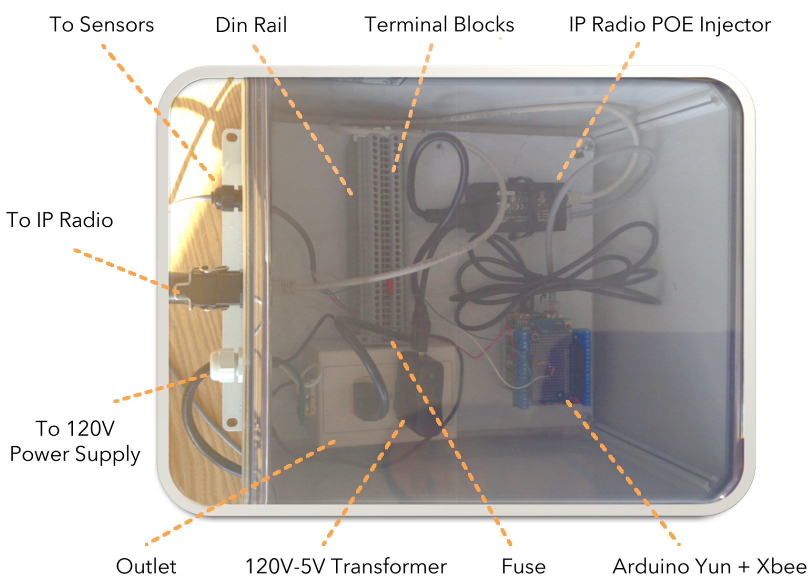

Figure 11: RTU of the SCADA System Prototype

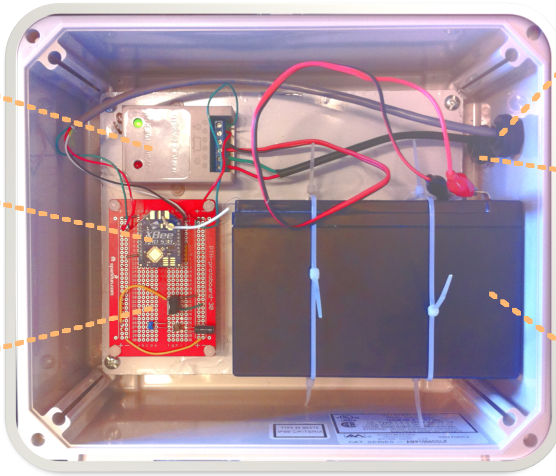

Figure 12: Wireless Sensing Part of the SCADA System Prototype

Table 5 shows the bill of materials for a single RTU that incorporates a NEMA 4 enclosure to help protect non-weatherproof equipment.

Table 5: Bill of Materials for a Single RTU

Item	Quantity	Unit Price ($)	Total Price ($)
Arduino Yun	1	72.00	72.00
Arduino Yun $120V - 5V$ Wall Charger	1	3.95	3.95
Arduino Yun USB Cable	1	5.49	5.49
Arduino Yun Wireless SD Shield	1	27.36	27.36
Proto-Screwshield (Wingshield) Kit	1	16.00	16.00
XBee Pro Series 2 (ZigBee Mesh)	1	40.95	40.95
Ubiquiti NanoStation	1	112.00	112.00
Enclosure	1	98.00	98.00
Aluminum Sub-Panel	1	23.00	23.00
Cable Gland	3	1.72	5.16
$120V$ Power Cable	1	1.50	1.50
Ethernet Cable	2	2.95	5.90
Ethernet Connector- Panel Receptacle	1	19.92	19.92
Ethernet Connector- Plastic Gland	1	10.16	10.16
Din Rail	1	5.74	5.74
Din Rail Outlet	1	30.84	30.84
Terminal Block	5	1.40	7.00
Fuse	1	0.21	0.21
Fuse Terminal Block	1	10.20	10.20
Stand Off	20	0.26	5.20
Break Away Header- Machine Pin	4	2.95	11.80
Break Away Header - Straight	2	1.50	3.00
Female Header	3	1.50	4.50
Resistor	5	0.45	2.25
Total			522.13

The RTU has been completed with a pre-installation cost of \$522.13, which is significantly cheaper than most commercially available products. A server in C12 Energy's North Dakota office acts as the MTU of the prototype, which has a physically connected IP-addressable radio and our software package installed.

The prototype testing results illustrate that the SCADA system design is practicable. Users can monitor and control the wellhead operation at C12 Energy's California office, thousands of miles away from the North Dakota oilfield, through virtual P2P connection with VPN. However, the wireless Ethernet/IP communication between RTU and MTU is not stable and thus needs further investigation. Possible reasons accounting for the instability include complicated geographical environments such as hills or mountains, and long distance. Future work might include attaching a high-gain antenna (HGA) to extend radio range and searching for more powerful radio options.

4.4 Sucker-Rod Pump Control Testing

Sucker-rod pump control shuts off the motor for a given time period when a well which has pumped off is detected based on down-hole conditions. An optimized control algorithm for sucker-rod pump control based on Everitt-Jennings algorithm[1] is implemented on the Arduino Yun. It is modified to deal with sucker-rod pumps which have multiple diameters at different positions. The inputs of the algorithm are up-hole rod force and displacement. To validate the performance of our algorithm, we gathered data from a producing well in C12 Energy's North Dakota oilfield. The artificial lift rod consists of a tapered rod string. The diameter data of the producing well are shown in Table 6.

Table 6: Lufkin C-456D-304-144 Pump Unit Data

1" Rods (ft)	7/8" Rods (ft)	3/4" Rods (ft)	1 1/2" K bars (ft)	Pony (ft)	Total Length (ft)
1750	1850	3700	150	14	7464

A full cycle of pumping takes an average of 8.635 seconds to complete four phases, including loading, upstroke, unloading, and downstroke. Data are preprocessed by linear interpolation to produce 360 data points at regular time intervals and broken into cycles of 8.635 seconds in duration with repeated entries removed. Figure 13 shows a plot of up-hole sucker-rod load versus time, and Figure 14 is a plot of up-hole displacement versus time.

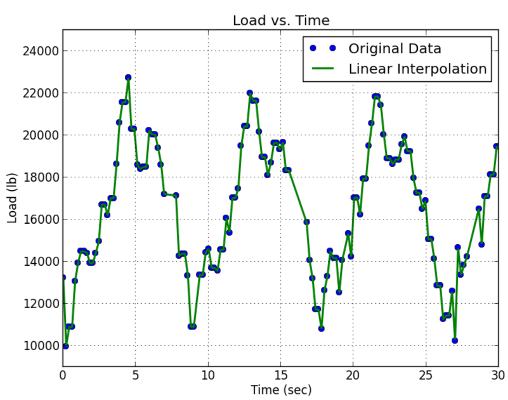

Figure 13: Load Versus Time

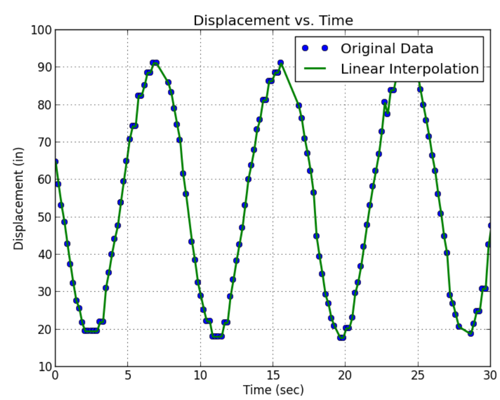

Figure 14: Displacement Versus Time

Figure 15 and 16 are the results of the algorithm. Figure 15 shows the accumulated downhole card under normal condition when the pump is operating normally from cycle No. 1 to cycle No.76. However, starting from cycle No.77, the pump is detected as having pumped off because the cycle's curve ceases to follow the shape of prior cycles, as shown in Figure 16. Arduino Yun will then output proper signals to shut off the motor and restart it after thirty minutes, when underground fluid reaches the pump capacity again. The testing results are completely correspond with the reality.

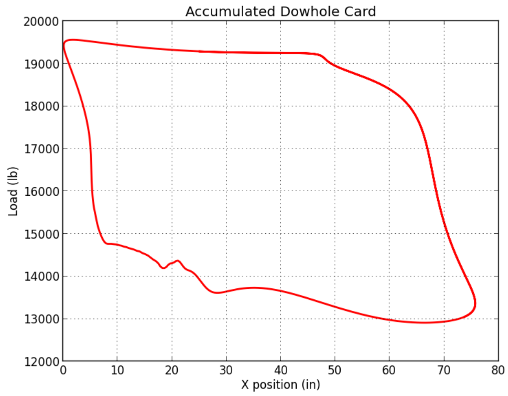

Figure 15: Cycle No.1-76: Normal Condition

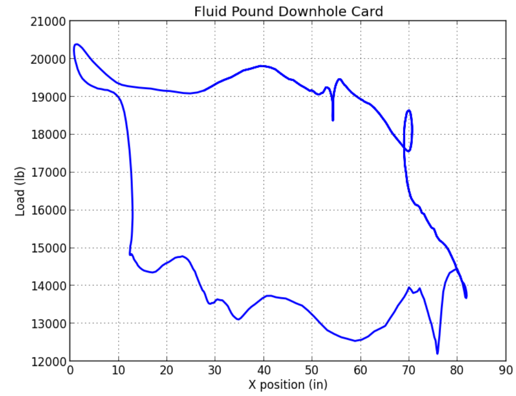

Figure 16: Cycle No.77: Pumped Off Condition

4.5 System Characteristics

Compared to other existing oilfield SCADA systems, our design has the following characteristics:

Efficiency Some widely-used SCADA systems are still centralized computing systems, meaning that RTUs do not have the ability to do calculations. Therefore, data storage, processing, and calculations have to be done on MTUs. Nevertheless, our RTUs have local single-board computers, Arduino Yuns, which can control the sucker-rod pump locally. This will reduce the potential risk of control dropout due to communication failure and eliminate the constraint on efficiency and performance caused by network speed, load, and throughput.

Over-the-Air Programmability OTA programmability means the possibility of distributing new software updates, configuration settings, or updating encryption keys to all RTUs remotely at one central location[45]. Thanks to the intelligent RTUs with Linux distribution and wireless Ethernet/IP radio communication, all RTUs in our SCADA system are remotely programmable, allowing engineers to remotely push software upgrades to every node in the network. This will increase convenience and greatly eliminate maintenance costs, especially for companies who have a huge number of RTUs distributed at various locations.

Low Cost The initial prototype of the RTU has been completed with a pre-installation cost of $522.13, not including the cost of sensors. The approximate post-installation cost should be around $1,000 per wellhead, which is significantly cheaper than most existing off-the-shelf SCADA systems. Additionally, accurate sucker-rod pump control will help reduce the operating cost and prolong the lifespan of sucker-rod pumps. This is undoubtedly meaningful for companies who want to scale SCADA system to hundreds of thousands of wells with low production rates.

Flexibility Most off-the-shelf SCADA systems have to to incorporate certain kinds of sensors and actuators. Typically they are not compatible with the existing sensors or actuators a company already has. However, our system is flexible enough to cooperate with almost all kinds of sensors and actuators with proper adjustment. Further, with appropriate programming, the same hardware can be used for tank level monitoring, process condition monitoring, and alarming. The high level of control and OTA programmability allows for efficient responses to dynamic oilfield conditions.

Reliability Interconnection between RTUs and MTU are not limited at the RTU-MTU level in our system. Instead, data sharing and forwarding are able to happen between different RTUs, increasing the reliability and redundancy of the system. It is not a point-to-multipoint (P2MP) communication but a mesh network. Data can always be successfully transferred through another path if one path is proven unavailable. With the help of the transmission control protocol (TCP), delivered data are reliable, ordered and error-checked[46].

Crash Recovery Several methods are involved in our system to offer crash recovery and error investigation. Wellhead sensor data are stored not only in the MySQL database on the central server but also in the SD cards of local single-board computers at the same time. Once the network crashes, all operation data are still being recorded and can be retrieved remotely when network recovers. Besides, every manual operation will be recorded in the Log relation, providing an opportunity to investigate reasons accounting for system failure.

Convenience Our web-based GUI makes it possible for users to use their web browsers to monitor and control the SCADA system wherever they are. It is not a stand-alone software that has to be installed on a certain kind of machine. People can use their laptops, tablets, or even mobile phones to access the system regardless of the operating system.

Robustness The SCADA system for CO_2 EOR is robust for outdoor use. Harsh outdoor environment has been taken into consideration when we choose system components. We require every part of our system to be weatherproof. The NEMA 4 enclosure will help protect non-weatherproof equipment against ingress of foreign objects like dirt, dust, rain, sleet, snow, and ice. It will also provide protection to personnel against access to hazardous parts.

5 Conclusions

SCADA systems are essential for oilfield operations. This paper proposes a SCADA system design that is efficient, flexible, convenient, cost-effective, robust, and reliable for CO_2 EOR. The design contains all the necessary parts to build a SCADA system, including hardware architecture and a software package. The system is highly competitive with existing SCADA systems and can be easily scaled to other industries with appropriate adjustment.

However, most of our design relies on other products, such as the single-board computer and wireless Ethernet/IP radio. In other words, the system's performance is dependent on the performance of its off-the-shelf components. Future work may include searching for substitutes of current components for better performance, stability and reliability. It is of great importance to keep in mind that we should never give up incorporating recent technological advances into our system. Besides, more in-field experiments and market research should be conducted before the design is finally commercialized.

We are hoping that this SCADA system design will help C12 Energy and other oil and gas companies participating in CO_2 EOR business to extract additional oil in a more efficient and cost-effective way.

References

[1] Thomas Aaron Everitt, JW Jennings, et al. An improved finite-difference calculation of downhole dynamometer cards for sucker-rod pumps. *SPE production engineering*, 7(01):121–127, 1992.

[2] Axel Daneels and Wayne Salter. What is scada. In *International Conference on Accelerator and Large Experimental Physics Control Systems*, pages 339–343, 1999.

[3] Bin Qiu and Hoay Beng Gooi. Web-based scada display systems (wsds) for access via internet. *Power Systems, IEEE Transactions on*, 15(2):681–686, 2000.

[4] Ronald E Terry. Enhanced oil recovery. *paper*, 137048:503–518, 2000.

[5] PL Bondor. Applications of carbon dioxide in enhanced oil recovery. *Energy Conversion and Management*, 33(5):579–586, 1992.

[6] Musse Mohamud Ahmed and WL Soo. Supervisory control and data acquisition system (scada) based customized remote terminal unit (rtu) for distribution automation system. In *Power and Energy Conference, 2008. PECon 2008. IEEE 2nd International*, pages 1655–1660. IEEE, 2008.

[7] GT Heng. Microcomputer-based remote terminal unit for a scada system. *Microprocessors and Microsystems*, 20(1):39–45, 1996.

[8] Il Moon. Modeling programmable logic controllers for logic verification. *Control Systems, IEEE*, 14(2):53–59, 1994.

[9] Joel Mobley, Brian M Cullum, Alan L Wintenberg, S Shane Frank, Robert A Maples, David L Stokes, and Tuan Vo-Dinh. Single-board computer based control system for a portable raman device with integrated chemical identification. *Review of scientific instruments*, 75(6):2016–2023, 2004.

[10] SA Boyer. Scada - supervisory control and data acquisition. *Instrument Engineers' Handbook, Volume Three: Process Software and Digital Networks*, page 357, 2007.

[11] Duo Li, Yoshizumi Serizawa, and Mai Kiuchi. Concept design for a web-based supervisory control and data-acquisition (scada) system. In *Transmission and Distribution Conference and Exhibition 2002: Asia Pacific. IEEE/PES*, volume 1, pages 32–36. IEEE, 2002.

[12] Sandip C Patel, Ganesh D Bhatt, and James H Graham. Improving the cyber security of scada communication networks. *Communications of the ACM*, 52(7):139–142, 2009.

[13] Mario Antonioletti, Malcolm Atkinson, Rob Baxter, Andrew Borley, Neil P Chue Hong, Brian Collins, Neil Hardman, Alastair C Hume, Alan Knox, Mike Jackson, et al. The design and implementation of grid database services in ogsa-dai. *Concurrency and Computation: Practice and Experience*, 17(2-4):357–376, 2005.

[14] Divyakant Agrawal, Sudipto Das, and Amr El Abbadi. Big data and cloud computing: current state and future opportunities. In *Proceedings of the 14th International Conference on Extending Database Technology*, pages 530–533. ACM, 2011.

[15] John Marcuse, Brad Menz, and Jeffrey R Payne. Servers in scada applications. *Industry Applications, IEEE Transactions on*, 33(5):1295–1299, 1997.

[16] Dennis J Gaushell and Wayne R Block. Scada communication techniques and standards. *IEEE computer applications in power*, 6(3):45–50, 1993.

[17] Rao Kalapatapu. Scada protocols and communication trends. *ISA EXPO*, 2004.

[18] Albert L Lederer, Donna J Maupin, Mark P Sena, and Youlong Zhuang. The technology acceptance model and the world wide web. *Decision support systems*, 29(3):269–282, 2000.

[19] Kwok-Hong Mak and BL Holland. Migrating electrical power network scada systems to tcp/ip and ethernet networking. *Power Engineering Journal*, 16(6):305–311, 2002.

[20] Xingzhen Bai, Xiangzhong Meng, Zhaowen Du, Maofa Gong, and Zhiguo Hu. Design of wireless sensor network in scada system for wind power plant. In *Automation and Logistics, 2008. ICAL 2008. IEEE International Conference on*, pages 3023–3027. IEEE, 2008.

[21] George W Irwin, Jeremy Colandairaj, and William G Scanlon. An overview of wireless networks in control and monitoring. In *Computational Intelligence*, pages 1061–1072. Springer, 2006.

[22] Tony Ciardiello. Wireless communications for industrial control and monitoring. *Computing & Control engineering journal*, 16(2):12–13, 2005.

[23] Vinay M Igure, Sean A Laughter, and Ronald D Williams. Security issues in scada networks. *Computers & Security*, 25(7):498–506, 2006.

[24] Wei Ye and John Heidemann. Enabling interoperability and extensibility of future scada systems. *University Southern California, USC/ISI Technical Report, ISITR-625*, 2006.

[25] Gabor Takacs. Power efficiency of sucker-rod pumping. *50th Southwestern Petroleum Short Course*, 2003.

[26] Doug Fitzpatrick. Method and system for controlling well bore fluid level relative to a down hole pump, November 14 1978. US Patent 4,125,163.

[27] Sam G Gibbs. Method for determining the pump-off of a well, April 20 1976. US Patent 3,951,209.

[28] SG Gibbs, AB Neely, et al. Computer diagnosis of down-hole conditions in sucker rod pumping wells. *Journal of petroleum technology*, 18(01):91–98, 1966.

[29] Roy M Knapp. A dynamic investigation of sucker-rod pumping. *MS thesis, V. of Kansas, Topeka*, 1969.

[30] Kun Li, Xian-wen Gao, Wei-bing Yang, Ying-long Dai, et al. Multiple fault diagnosis of down-hole conditions of sucker-rod pumping wells based on freeman chain code and dca. *Petroleum Science*, 10(3):347–360, 2013.

[31] Robert George Smith, Mike Hall, Andrew John Franklin, Espen S Johansen, Omer Haldun Unalmis, et al. Field-wide deployment of in-well optical flowmeters and pressure/temperature gauges at buzzard field. In *Intelligent Energy Conference and Exhibition*. Society of Petroleum Engineers, 2008.

[32] Bernardo Ordonez, Andres Codas, Ubirajara Franco Moreno, and Alex Teixeira. Sucker-rod pumping system: simulator and dynamic level control using bottom hole pressure. In *Emerging Technologies and Factory Automation, 2008. ETFA 2008. IEEE International Conference on*, pages 282–289. IEEE, 2008.

[33] CG Guffey, JD Rogers, LR Hester, et al. Field testing of variable-speed beam-pump computer control. *SPE Production Engineering*, 6(02):155–160, 1991.

[34] Jon Seaberg, Jin Yan, and James Zhuge. Web based user interface, December 24 2003. US Patent App. 10/743,880.

[35] Goran Zecevic. Web based interface to scada system. In *Power System Technology, 1998. Proceedings. POWERCON'98. 1998 International Conference on*, volume 2, pages 1218–1221. IEEE, 1998.

[36] Omar H Alhazmi and Yashwant K Malaiya. Measuring and enhancing prediction capabilities of vulnerability discovery models for apache and iis http servers. In *Software Reliability Engineering, 2006. ISSRE'06. 17th International Symposium on*, pages 343–352. IEEE, 2006.

[37] M Cioca, L-I Cioca, and S-C Buraga. Spatial [elements] decision support system used in disaster management. In *Digital EcoSystems and Technologies Conference, 2007. DEST'07. Inaugural IEEE-IES*, pages 607–612. IEEE, 2007.

[38] Peter Membrey, Eelco Plugge, and Tim Hawkins. *The definitive guide to MongoDB: the noSQL database for cloud and desktop computing*. Apress, 2010.

[39] Yao-Wen Huang, Shih-Kun Huang, Tsung-Po Lin, and Chung-Hung Tsai. Web application security assessment by fault injection and behavior monitoring. In *Proceedings of the 12th international conference on World Wide Web*, pages 148–159. ACM, 2003.

[40] Arduino yun introduction. `http://arduino.cc/en/Main/ArduinoBoardYun?from=Main.ArduinoYUN`, accessed April 13, 2014.

[41] Charles Bell. Mysql and arduino: United at last! In *Beginning Sensor Networks with Arduino and Raspberry Pi*, pages 223–265. Springer, 2013.

[42] Ubiquiti nanostation introduction. `http://www.ubnt.com/nanostation`, accessed April 13, 2014.

[43] Radha Guha. Toward the intelligent web systems. In *Computational Intelligence, Communication Systems and Networks, 2009. CICSYN'09. First International Conference on*, pages 459–463. IEEE, 2009.

[44] Daniel Barrett, Richard Silverman, and Robert Byrnes. *SSH, the secure shell: the definitive guide*. " O'Reilly Media, Inc.", 2005.

[45] Over-the-air programming. `http://en.wikipedia.org/wiki/Over-the-air_programming`, accessed April 13, 2014.

[46] M Janani and P Chenna Reddy. International journal of networks and systems. *International Journal*, 2(6), 2013.