

Software Approaches to Support RFID Temperature Monitoring for Blood Products

Zi Lin

A dissertation submitted to Auckland University of Technology in partial fulfilment of
the requirements for the degree of Master of Computer and Information Sciences

March 2015

School of Computing and Mathematical Sciences

Primary Supervisor: Assoc. Prof. Dave Parry

Secondary Supervisor: Dr. John Ayoade

Table of Contents

List of Figures.....	i
List of Tables.....	ii
Appendices	iii
Acronyms.....	iv
Attestation of Authorship.....	v
Acknowledgement.....	vi
Abstract	1
Chapter 1 Introduction.....	2
1.1 Research Rationale	3
1.2 Research Objectives	4
1.3 Research Structure	5
Chapter 2 Literature Review.....	6
2.1 Overview of RFID Technology.....	6
2.1.1 Brief History of RFID	7
2.1.2 RFID System Architecture.....	7
2.1.3 Equipment of RFID System	9
2.1.4 Frequency Range of RFID System.....	11
2.2 RFID Temperature Sensor Tag.....	12
2.3 RFID Usage in Drug Tracking.....	15
2.4 Prior Implementation of RFID Blood Tracking Systems.....	18
2.5 Safety of Blood Tracking using RFID Technology.....	20
Chapter 3 Research Methodology.....	21
3.1 Design Science	22
3.2 Research Model.....	24

Chapter 4 System Demonstration	28
4.1 Overview.....	28
4.1.1 Architecture.....	30
4.1.2 Technologies.....	32
4.2 System Model.....	37
4.2.1 Use Case Diagrams	38
4.2.2 Workflow Diagram	43
4.2.3 Class Diagram	44
4.2.4 Database Diagram	46
4.3 GUI Diagrams.....	47
4.3.1 Home/Login Page	47
4.3.2 Main Menu Page	48
4.3.3 Temperature Dashboard Page	49
4.3.4 Temperature Data Page	50
4.3.5 Blood Bag Data Page	51
4.3.6 Temperature Report Page.....	52
4.4 Key Functions of the Prototype Application.....	53
4.4.1 Temperature Upload Function/Simulator function	53
4.4.2 Dashboard Displaying Function.....	56
4.4.3 Alarm Function	61
4.4.4 Generating Temperature History Report Function	62
4.5 Testing	64
Chapter 5 System Evaluation.....	65
5.1 Simulation Process	65
5.1.1 Preparation Work	66
5.1.2 Real-time operation.....	67

5.1.3 Post Operation.....	69
5.2 Summary.....	70
Chapter 6 Discussion	71
6.1 Summary.....	71
6.2 Benefits.....	73
6.3 Limitations	75
Chapter 7 Future Work.....	77
Chapter 8 References	80

List of Figures

Figure 2.1 A high-level view of the interaction between reader and tag in the RFID auto-identification process (Roussos, 2008).....	8
Figure 2.2(From left to right) portals, tunnels, handhelds, forklift readers and smart shelves (Glover & Bhatta, 2006).....	10
Figure 2.3 Top Block Diagram of an RFID temperature sensor (Martinez Brito & Rabaeijs, 2013)	14
Figure 3.1 Research Model.....	24
Figure 4.1 Architecture of Proposed Temperature Monitoring System	30
Figure 4.2 Use Case Diagram of User Account Management System.....	38
Figure 4.3 Use Case Diagram of Temperature Data Management System.....	39
Figure 4.4 Use Case Diagram of Blood Bag Management System	40
Figure 4.5 Use Case diagram of Real-Time Display System.....	41
Figure 4.6 Use Case Diagram of Reporting System	42
Figure 4.7 Workflow Diagram of Proposed Blood Tracking Solution	43
Figure 4.8 Class Diagram of Prototype Application.....	44
Figure 4.9 Database Diagram of Prototype Application.....	46
Figure 4.10 User Interface of Login Page	47
Figure 4.11 User Interface of Main Menu Page	48
Figure 4.12 User Interface of Temperature Dashboard Page	49
Figure 4.13 User Interface of Temperature Data Page	50
Figure 4.14 User Interface of Blood Bag Data Page	51
Figure 4.15 User Interface of Temperature Report Page.....	52
Figure 5.1 Workflow Diagram of Preparation Work	66
Figure 5.2 Workflow Diagram of Real-Time Operation.....	67
Figure 5.3 Workflow of Post Operation	69

List of Tables

Table 2.1 Comparison of Different Tags (Schuster & Brock, 2007)	9
Table 2.2 RFID frequency range (Rundh, 2008).....	11
Table 4.1 Version of Selected Tools	32

Appendices

Appendix A – Sample Temperature Report..... 90

Appendix B – Sample Temperature Worksheet 97

Acronyms

AJAX	Asynchronous Javascript and XML
API	Application Programming Interface
CMOS	Complimentary Metal-Oxide Semiconductor
DSR	Design Science Research
EPC	Electronic Product Code
GPRS	General Packet Radio Service
GSM	Global System for Mobile communications
GUI	Graphical User Interface
HF	High Frequency
HIS	Hospital Information System
IDE	Integrated Development Environment
IIS	Internet Information Services
LW	Low Frequency
PDF	Portable Document Format
RF	Radio Frequency
RFID	Radio Frequency Identification
ROI	Return on Investment
SAW	Surface Acoustic Wave
UHF	Ultra High Frequency
UI	User Interface
UML	Unified Modelling Language

Attestation of Authorship

I hereby declare that this submission is my own work that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgements), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Sincerely

Zi Lin

Acknowledgement

I am very pleased to acknowledge the people for their given assistance and encouragement throughout my thesis.

First and foremost, I am very grateful to my primary supervisor, Assoc. Prof. Dave Parry, who guided me along the way to achieve this thesis. He provided constructive thoughts and critiques with regard to my work, as well as mental encouragement and intimate communication. Without the supervision and assistance from him, this thesis would not have been carried out.

Secondly, I would like to thank my secondary supervisor, Dr. John Ayoade, who analysed problems and brought up new ideas in meetings.

Thirdly, I greatly appreciate the hospital staff from North Shore Hospital for their professional advice. After conducting several meetings with them, they expanded my knowledge and vision in the field of RFID applications in hospital. I was also invited to their blood bank and the staff demonstrated the hospital equipment used for blood transfusion, including blood bag and transfusion container.

I would like to thank my classmates and friends from Auckland University of Technology for their assistance and encouragement. Their support reduced my anxiety and I became less stressful in writing the thesis.

Finally, I heartily appreciate the support of my mother, who took care my living and encouraged me during my thesis.

Abstract

Blood transfusion has always been crucial and strictly monitored. Currently, there are many standards for managing transfusion blood bags. One standard is maintaining the temperature of blood bags during transfusion. Transfusion staff are expected to check the temperature of the transfused blood bag in real time and take action if the temperature exceeds the limit. However, many healthcare organizations have not implemented procedures by which transfusion staff can instantly track the temperature of a transfused blood bag. Therefore, a temperature monitoring solution that displays the temperature data in real time is necessary. In this solution, radio-frequency identification (RFID) technology is used to actively detect the temperature of the blood bag and upload the temperature record to the web server.

This research was carried out to analyze the benefits and limitations of a web-based RFID temperature monitoring application. The first chapter is a literature review on RFID technology, application of RFID drug tracking, and prior research into blood transfusion and safety concerns in using RFID on blood products.

Next, a prototype was created by following a design science approach, and was evaluated in simulations and a review of the simulation process. The potential benefits and limitations of the prototype application are discussed in detail. Finally, future developments and researches stemming from the present work are suggested.

Chapter 1 Introduction

Blood management has become more strictly monitored than in the past, and the relevant technologies have advanced with increasing global safety demands. The safety of patients is very important in blood transfusion. To minimize the risk to patients, all aspects related to medical security and health products should be top priority (Abarca, Fuente, Abril, García & Pérez-Ocón, 2009). RFID is one of the technologies used for satisfying these safety demands. Over the past few years, RFID blood tracking systems have usually received positive feedback, even when used for general identification purposes only. RFID systems would be improved by enhancing the visibility of the blood product, enabling unambiguous tracking from the origin to the target of the blood product ("RFID advance to improve safety", 2013).

Currently, RFID technology can track more information of blood bags (such as temperature and location) when cooperating with multiple devices and software. Many district health boards also impose strict standards for maintaining the temperature of red blood cells (RBCs). For instance, if RBC blood units are exposed to external environment outside of fridge for more than 30 minutes, they should not be allowed to return to blood fridge and must be abandoned if their temperature is greater or equal to 10° C (Grandmont et al, 2014). Hess and Grazzini (2010) assumed that RBCs become damaged if the temperature of the blood bag exceeds the approved 1–10° C transport storage temperature. Therefore, we aspire to build a small-scale prototype application for monitoring the temperature of blood bags during blood transfusions.

This research proposes a web-based solution by which healthcare organization administrators can remotely identify and retrieve the temperature data of transfused blood bags using RFID technology. A prototype of the temperature monitoring application is carried out to demonstrate and analyze its benefits and issues. Section 1.1 presents the rationale and motivation behind this research. Section 1.2 defines the research questions and research objective. The overall structure of this research report is presented in Section 1.3.

1.1 Research Rationale

RFID has significantly progressed throughout the past decade. The most important feature is the cost effectiveness of setting up an RFID system, which is always better than other identification methods such as barcodes. RFID technology not only reduces the cost of drug storage, but it also improve the safety and efficiency features of medical product transfusion (“Tracking donated blood”, 2011). Most of the proposed RFID blood tracking systems involve return on investment (ROI) analysis. For example, an analysis of the base case (a mid-size blood center collecting 225,000 units/year) predicted a positive net present value of \$83,560 (11.2% return on investment) over a five-year planning horizon, resulting in an approximate pay-back period (time to recover the investment) of four years (Davis, Geiger, Gutierrez, Heaser & Veeramani, 2009).

The demands of blood product management have become increasingly complex and composite. Of particular importance is monitoring the blood bags while cooperating with other sensor devices. This involves “identification and selection of prospective blood donors, adequate collection of blood and preparation of blood components, quality laboratory testing and ensuring the safest and most appropriate use of blood/blood components” (Bocchi & Giacomo, 2007). RFID technology can overcome many problems associated with other identification methods. For example, RFID technology can detect a group of blood bags at the same time, and thereby rapidly collect their information. Manually scanning the barcode of each blood bag is much more time-intensive. An advantage of RFID technology is its digital nature; an RFID reader can technically cooperate with other electronic devices such as temperature sensors.

However, some places have been slow to embrace RFID blood tracing systems. The situation is even worse for complex systems applying active tags, which have been rarely deployed during the past 4–5 years. The likely reason for this resistance was suggested by a transfusion medicine market manager, who posed the question “Do the expected safety improvements and cost benefits justify the costs?” To clarify, he added “While RFID tags are expected to become less costly as the worldwide supply increases, in the short term the premium to be paid for a smart label is significant” (Wray, 2011) Although RFID systems eventually return a positive outcome, the initial investment of a set of RFID equipment remains considerable to small organizations (e.g. private hospitals). Therefore, reducing the setup cost of RFID equipment and relevant resources is a present mission in promoting RFID blood management systems.

Currently, many places lack RFID temperature monitoring systems for blood transfusions. For example, active RFID tags are scarcely used in New Zealand's hospitals and healthcare organizations. Thus far, RFID tags are used only to assign a unique identity to each blood bag, similar to a barcode. The temperature status during blood transfusion remains unrecognized. The blood quality cannot be guaranteed if transfusion staff lack real-time temperature details of the transfused blood bag. Such missing information could indirectly compromise the health of patients. Thus, to satisfy the latest temperature requirements of blood transfusions, a small-scale low-cost RFID temperature monitoring solution is urgently required.

1.2 Research Objectives

This report proposes and aims to answer three research questions:

1. What components are required to build a web-based system for monitoring the temperature of blood bags?
2. What tasks could be done by blood transfusion staff once the temperature monitoring application becomes available?
3. What are the potential benefits of having a temperature monitoring system in blood transfusions?

The research proposes a prototype of a temperature monitoring application, and assesses its potential benefits and drawbacks for blood transfusion. The first research question is to determine the essential components of a temperature monitoring system. The second question will address the functions of the prototype application. The third question will be answered by evaluating and analyzing the temperature monitoring application.

1.3 Research Structure

The main body of this thesis is divided into six chapters. Chapter 1 has presented the rationale and motivation behind the present research, and stated the objectives in the form of three research questions.

Chapter 2 is a literature review on RFID technology. It introduces the history of RFID, the architecture of a RFID system, and the RFID equipment and frequency range. RFID temperature sensors are also discussed. Then, we overview the use of RFID technology in identifying and tracking drug administered by healthcare organizations. Blood tracking is one component of drug tracking. Next, several implemented or proposed blood tracking systems are reviewed. Finally, Chapter 2 discusses the safety of blood products evaluated by RFID technology.

Chapter 3 introduces the methodology of this research. Since the design science approach is pivotal to this research, it is reviewed and discussed in detail. Each stage of the proposed research model is then explained.

Chapter 4 details the essence of the prototype application. To demonstrate how the prototype works in real blood transfusion, the entire temperature monitoring solution is outlined. The tasks and user interfaces are described by system models and GUI diagrams. Several key functions are also explained in detail, along with snippets of the developed code. The results and errors during testing of the system are also presented.

Chapter 5 assesses the performance of the prototype application by simulating several envisaged scenarios. The tasks in each scenario are explained, and the simulations are reviewed by descriptive analyzes.

Chapter 6 states the accomplishments of the research. The benefits and limitations of the prototype application are then discussed, respectively.

Chapter 7 proposes ideas for future research and development. Namely, we need to improve the current functionality of the prototype application and solve several identified problems. Ideas for solving potential issues of the application are also suggested.

Chapter 2 Literature Review

The literature review is divided into several sections reviewing different aspects of the literature. Section 2.1 is dedicated to RFID technology. Section 2.2 presents RFID temperature sensor and several common technologies used for the sensors. Section 2.3 introduces application in RFID drug tracking. The reason to review RFID-based drug tracking is because blood tracking is essentially a type of drug tracking. Thus, we acquire a bigger picture of RFID applications in this area. Section 2.4 reviews previous blood transfusion systems using RFID technology, and Section 2.5 discusses how RFID technology influences the safety of blood products. The review of previous blood transfusion systems is based on overviewing; the aim of the review is to understand the previous achievements instead of identify the flaws of previous systems. The review helps in identifying the strengths and potentials of the proposed application, which have not been seen in or more beneficial than the previous blood transfusion systems.

2.1 Overview of RFID Technology

RFID is a terminology encompassing several information and communication technologies that automatically identify objects, locations, and individuals to computing systems with the cooperation of radio frequency (Roussos, 2008). The RFID system uses its RFID reader to detect the RFID tag of a remote object. Characteristically, the RFID reader transmits the data through radio signals and simultaneously provides a power source. In this way, RFID tag is powered by wirelessly transmitted radio frequency power and does not require internal power source to be functional (Roussos, 2008). The internal power sources of semi-active and active RFID tags are reserved for other purposes, such as initiatively signaling the RFID reader and extending the detection range between the reader and the tag.

The remainder of this section consists of 4 parts. Section 2.1.1 introduces the history of RFID technology, and Section 2.1.2 explains the fundamental structure of a RFID- based system. Section 2.1.3 illustrates the essential equipment of an RFID system. The frequency range standards of RFID technology are briefly introduced in section 2.1.4.

2.1.1 Brief History of RFID

Guglielmo Marconi presented the first successful remote transmission across the Atlantic using radio frequency technologies in the year 1896 (Landt, 2005). His research was the first attempt to transmit data using radio frequencies. In 1906, Ernst F.W. The demonstration includes “the first continuous wave (CW) radio generation and transmission of radio signals” (Landt, 2005). RFID was developed in the 20th century, and was initially used in radar. The fundamental concept of radar is that two objects can communicate via radio frequency transmission. The military highly valued the potential of radar application, so they conducted a series of classified radar development during the two world wars (Landt, 2005). In 1948, a study regarding the pilot exploration of FRID named “Communication by Means of Reflected Power” was published by Harry Stockman (Stockman, as cited in Ko, 2009). Nowadays, RFID technology has become more generic. Most systems that identify remote objects by radio frequencies are now called FRID systems.

2.1.2 RFID System Architecture

Kabachinski (2005) showed that RFID system has three main parts: the reader (interrogator), the antenna and the tag (transponder). The reader is the device that reads data from the tag (Kabachinski, 2005). The antenna is designed to “generate a magnetic field which activates the magnetic tag and enable communication between the tag and the transponder” (Rundh, 2008). In most case, the antenna and transponder are combined as a RFID reader to minimize the components needed in a RFID system (Rundh, 2008). Handheld RFID reader is a typical example of merging both antenna and transceiver on one device. However, sometimes external antenna needs to be used for increase the detection range. Built-in antenna usually has limited detection capacity due to its small size.

The RFID tag stores data in an electrical circuit (Rundh, 2008). The most common information stored on RFID tag would be EPC code. The EPC is “a sort of license plate with a unique number which allows to clearly and individually identify a marked object” (Finkenzeller & Müller, 2010). The EPC establishes a protocol between GS1’s barcode-based identifiers and RFID, allowing the identification interactively supported (“Electronic Product Code,” n.d.). It can be seen that EPC could be used to identify object within RFID systems. *EPC/RFID Certification* (n.d.) shows that a wide range of UHF devices are already certified under EPC/RFID standards.

Figure 2.1 displays the essential components of a RFID system and the transmission process of radio frequencies. The RFID reader communicates with RFID tags via its antenna. Basic transmission is a three-step process. First, the reader supplies electric power to the RFID tag by transmitting a radio frequency. The reader then issues tag commands to the tag and the tag sends radio signals back to the reader for response.

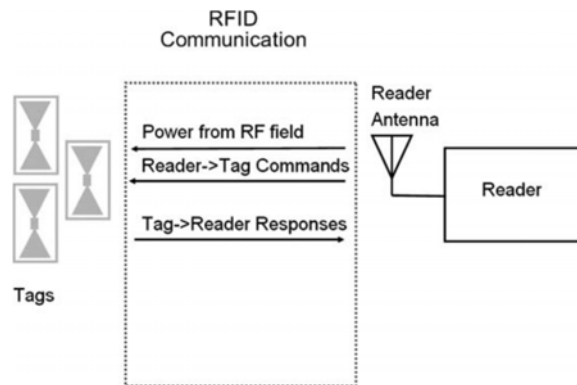


Figure 2.1 A high-level view of the interaction between reader and tag in the RFID auto-identification process (Roussos, 2008)

2.1.3 Equipment of RFID System

Two essential components of a RFID system are the RFID tag and RFID reader. Generally, the RFID tag is attached to an object requiring identification. The tag is read (and the object identified) by the RFID reader. The RFID tag and RFID reader are explained below in detail.

RFID Tag

RFID tags enables remote continuous identification of product while it travels through the supply chain (Schuster & Brock, 2007). Essentially, a set of goods can be pasted with RFID tags and tracked anytime thereafter simply by moving the reader. This method is more efficient and effective than methods such as barcodes, which demand a clean environment between the product tag and detection device.

There are many advantages of RFID tags. Passive tags require only several kilobytes of memory, but record complex identification details that cannot be recorded by traditional identification methods. Furthermore, there are potential RFID tag integration with electrical product, including camera, cell phone and family entertainment devices (Schuster & Brock, 2007), via a customized program that connects the tags with electronic devices.

Currently, many models of RFID tags are available in the market. Tags are categorized into active, semi-passive and passive. Active tags provide high functionality support but are usually the most expensive, whereas passive tags tend to be more cost-efficient. Some specifications of these tags are provided in Table 2.1.

	Active	Passive	Semi-Passive
Power Source	Battery	Induction from electromagnetic waves emitted by reader	Battery and Induction
Read Distance	Up to 30 meters	3–7 meters	Up to 30 meters
Proximity Information	Poor	Good	Poor
Frequency Collision	Hi	Medium	Hi
Information Storage	32 kb or more. Read/Write	2 kb Read only	32 kb or more. Read/Write
Cost/Tag	\$5 – \$100	10 c ^a	Under Development, Some commercial applications

Table 2.1 Comparison of Different Tags (Schuster & Brock, 2007)

RFID Reader

There are many different shapes, sizes and protocol support for RFID readers. However, it must meet a standard requirement; that is, “a particular reader may be acceptable for an application in one region of the globe but not in another” (Glover & Bhaṭṭa, 2006). RFID readers are distinguished by three factors: 1) Shape and size, 2) Standards and protocols and 3) Regional differences. The shape and size of a reader determines whether the reader will be embedded in a handheld device or remain stationary. The standards and protocols refer to the settings of RFID readers. These settings control whether readers may read one or more tags, or are limited to reading tags from the same manufacturer. Since many readers can be re-programmed, standards and protocols are highly customizable. Regional differences refer to the reader specifications, including the frequency and power levels.

RFID readers may be portals, tunnels, handheld devices, forklift readers or smart shelves. Each of these forms is applicable to different situation. For example, portal readers could be used at the gate to detect the passing products. The diverse structures of RFID readers are illustrated in Figure 2.2.

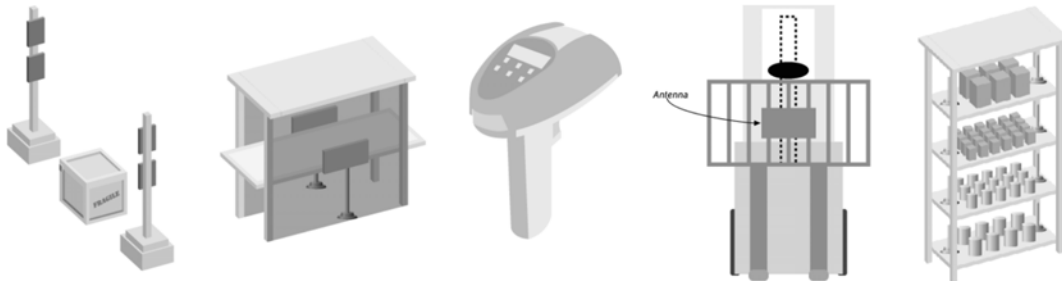


Figure 2.2(From left to right) portals, tunnels, handhelds, forklift readers and smart shelves (Glover & Bhaṭṭa, 2006)

There were a lot of RFID systems and studies using handheld RFID readers in the past (Bang, Chinzorig, Koh, Cha, & Ahn, 2012; Chen & Huang, 2013; Gupta, Singh, Bal, Kedia, & Harish, 2014; Huang & Hsu, 2012; Lin, Lee, Pan, & Chen, 2013; Miesen, Kirsch, & Vossiek, 2013; Zhu, Jiang, Zhang, & Guan, 2014). Handheld RFID reader can include GPRS, Wi-Fi and Bluetooth components (“Chainway C3000” n.d.). GPRS is a mobile data service whereas Wi-Fi and Bluetooth are used for short range wireless communication. It means that handheld RFID reader is able to connect to the Internet and transmit data in real time.

2.1.4 Frequency Range of RFID System

The reading range of a RFID system depends on the distance between a tag and the reader (Rundh, 2008). The detection rate is reduced at the edge of the maximum reading range. If the reader and tag separate beyond the reading range, the tag becomes unreadable. The reading range also depends on the type of RFID tag. Many of the active tags are superior in read–write capabilities, memory capacity and detection distance comparing to passive tags (Rundh, 2008).

It should be pointed out that the detection environment can also influence the reading performance. For instance, humidity does not affect the bar code reading performance, but can dramatically decrease the detection range for RFID tags (Schuster & Brock, 2007).

Common radio frequency standards are summarized in Table 2.2. Listed are the frequency ranges, read ranges from passive tags and sample applications of each frequency standard.

Name	Frequency range	Read range for passive tags	Application
LW	30-300 Khz	50 cm	Access control; animal identification; close reads of items with high water content
HF	3-30 Mhz	3 m	Building access control, smart cards; airline baggage tracking; electronic article surveillance
UHF	300 Mhz-3 Ghz	9 m	Item management, e.g. boxes and pallets
Microwave	>3 Ghz	>10 m	Vehicle identification

Table 2.2 RFID frequency range (Rundh, 2008)

2.2 RFID Temperature Sensor Tag

In order to remotely monitor the temperature of tagged product, a feasible solution would be implementing wireless communication between temperature sensors and RFID tag to indirectly transmit the temperature data to RFID reader (Opasjumruskit et al., 2006). Kang et al (2013) suggests that RFID temperature sensor is helpful and efficient in detecting items at a distance. RFID enabled temperature sensing solutions not only improves the efficiency of regular RFID systems, and it is able to integrate some applications which was not commonly implemented due to efficiency issues, such as the cold-chain logistics, the storage of temperature-sensitive products, and medical diagnostic tests and procedures (Kang et al., 2013). There were various studies and development of RFID temperature sensor tags in the past few years. Two common technologies used to build temperature sensor are SAW and CMOS.

SAW sensor was actually applied more than only temperature detection in many industries. SAW technology was known to remotely monitor information of product, including temperature, pressure, chemical compositions, vapour, humidity, torque and others mechanical constraints (Wang, 2014). SAW sensors are usually cooperated with various wireless, passive multi-sensor applications. These applications are identified as “small, rugged, radiation hard, and offer a wide range of material choices for operation over broad temperature ranges” (Saldanha & Malocha, 2012). Saldanha and Malocha (2012) discussed that SAW sensors could work in an environment up to 200 Celsius, so it can be applied in a wide range of industries. In contrast to the integrated circuit based RFID combined using simulation sensor, SAW RFID is more superior at piezoelectric effects, recognized as “an essentially passive and harsh-environment-hard technology and has become the front runner of the chipless RFID tag technologies” (Plessky & Reindl, as cited in Kang et al, 2013). There are also more SAW-RFID temperature sensor studies (Barton et al., 2010; Fachberger, Binder, & Erlacher, 2009; Gallagher & Malocha, 2013; Kozlovski, Malocha, & Weeks, 2011; Reindl, Pohl, Scholl, & Weigel, 2001; Rodriguez et al., 2014).

Applying CMOS technology is another approach to design RFID temperature sensor. The most significant advantage of using CMOS technology is low power consumption. Using a CMOS process, we can also incorporate signal-interfacing circuitries (for example, a current-to-voltage converter, an analogue-to-digital converter, and so on) within a single chip to minimize the system’s power consumption, cost, and size (Opasjumruskit et al., 2006). Mohamad, Fang,

Amira, Bermak and Benammar (2013) conducted a research on temperature sensor based on ring oscillator. Ring oscillator is a device on electric circuit of temperature sensor. The ring oscillator uses the CMOS thyristor delay element and has an extremely low power consumption of about 47nW at room temperature with a supply of 0.5V (Mohamad et al, 2013). Mohamad et al (2013) then proceeded to simulation test by using Chartered Semiconductor's 0.18 μ m technology. The experiment result indicated that CMOS thyristor delay element enables low power operation under various circumstances (Mohamad et al, 2013). More studies was done in regard to CMOS temperature sensor using RFID technology (Jun et al., 2010; Kocer & Flynn, 2006; Law, Bermak, & Luong, 2010; Martinez Brito & Rabaeijs, 2013; Zito, Aquilino, Fragomeni, Merenda, & Corte, 2010).

There are studies into the design of algorithm and architecture of temperature sensor. For instance, Martinez Brito and Rabaeijs (2013) presented four temperature sensor schemes based on CMOS technology. They firstly identified a high-level architecture of RFID temperature sensor tag as shown in Figure 2.3 and proposed four different architectures for temperature sensor. Then, they conducted a case study to test temperature data based on the most superior architecture. The result of case study showed that the measured data showed high consistency and linearity (Martinez Brito & Rabaeijs, 2013). Zhang, Li, Gao and Li (2014) proposed a with dual-label design. The design scheme is to extract the core part (temperature sensitive component) of temperature sensor, and integrate with RFID tag technology in an innovative form (Zhang et al, 2014). The experiment conducted by Zhang et al (2014) showed that the dual-tag design could reduce production cost comparing with previous technology integration scheme.

The proposed system is different from the SAW and CMOS methods that only the movement in and out of blood bank is detected. However, these methods are very valuable in studying how blood bags are been monitored in current state of blood transfusion.

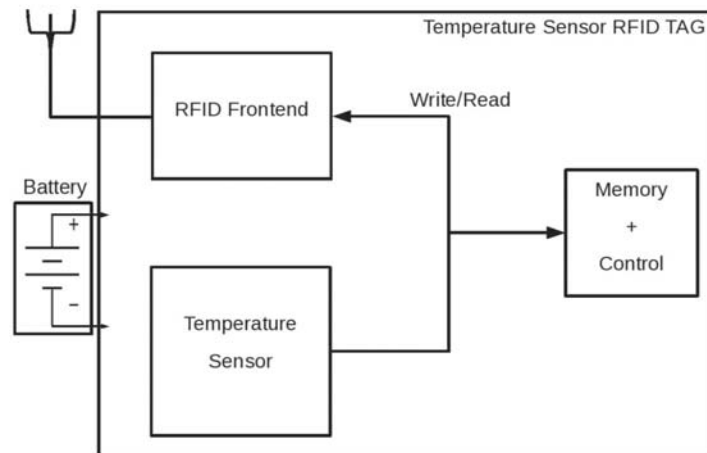


Figure 2.3 Top Block Diagram of an RFID temperature sensor (Martinez Brito & Rabaeijs, 2013)

There are also some other research in RFID temperature sensor. Girbau, Ramos, Lazaro, Rima and Villarino (2012) proposed a time-coded chipless RFID temperature sensor tag. The method to identify sensor tag is to alter the duration of delay time period (Girbau et al, 2012). The delay line is a specially designed circuit on the temperature sensor and used for identification purpose. Lorenzo, Girbau, Lázaro and Villarino (2014) demonstrated a frequency-coded RFID temperature sensor tag based on dual-band resonator. The communication between sensor tag and antenna is conducted by emitting two resonances. The first respond is the communication verifying the identification of sensor tag and the second respond contains temperature data information (Lorenzo et al, 2014). In order to maximize the sensor performance, the second respond is adjusted with “a varactor con-trolled by a negative temperature coefficient sensor and its low-power consumption conditioning circuit” (Lorenzo et al, 2014).

2.3 RFID Usage in Drug Tracking

The error rate and safety of medications are the most critical factors in drug tracking. These two factors are discussed below along with some previous achievements from previous studies.

In the healthcare industry, the most common and critical problem is medication error (Kohn, Corrigan, & Donaldson, 2002). It is defined as a failure in the treatment process if the patient is actually or potentially harmed by the error. It can occur at different stages of the treatment, such as prescription, dispensation and administration (Aronson, 2009). Severe medication errors can have fatal results. The chance of causing deaths by medication errors increased “approximately 10 times in UK, from 20 to near 200 between 1990 and 2000” (from 20 to ~200; Audit Commission, 2001). It was predicted that between 44,000 to 98,000 people died in hospitals annually, because they were mistakenly given wrong medication which should not have happened (Charatan, 1999; Kohn, Corrigan, & Donaldson, as cited in Ajami & Rajabzadeh, 2013).

Although medication errors are almost unavoidable, their frequency can be reduced by adopting appropriate information technology systems. RFID can potentially enhance patient safety, since it provides an automatic identification method for tracking the movement of medicines at the item-level (Acierno, Maffia, Mainetti, Patrono, & Urso, 2011; Catarinucci, Colella, De Blasi, Patrono, & Tarricone, 2012).

Becker et al. (2009) proposed a RFID-based Smart Drawer that monitors how patients take their medications. The Smart Drawer determines whether a patient’s medicine bottles are inside or outside the drawer. It also records the times of opening and shutting the drawer. These activities are uploaded to the database with a timestamp. The RFID reader is placed at the bottom of the smart drawer while the antenna is placed between the reader and the drawer. The medicine bottles are locatable because each bottle is attached with a RFID tag for identification purposes. Therefore, the system can detect whether the patient takes the right dose of the right medicine at the right time.

Access points with different functions and user interfaces for three user groups (patient, caregiver and maintainer) have been developed. User interfaces designed for patients alert the patient when he does not follow the predefined medicine-taking procedure. Patients can also search historical data for information. For example, they can search for the types of drugs they

have already taken, and instructions for later medicines. The caregivers' main function is to obtain the history of patients taking the medicine. This group is authorized to enter and modify the prescriptions. Hospital staff are allowed to change the system functionality and export the stored data for future analysis.

Similarly, McCall, Maynes, Zou, & Zhang (2010) proposed the RFID-based Medication Adherence Intelligence System (RMAIS), expediently designed for home use by aged patients or people taking complex prescriptions. This system comprises a RFID reader, a scale, a microcontroller, an LCD panel, and a motorized rotation platform. Part of the platform is cut off and directly attached on top of the scale. By rotating the spoke, the medicine bottle is pushed and moved from the main platform to the scale-top plane section (the scale-top plane refers to the complete scale device). Thus, the weight of the bottle can be accurately measured to determine whether the patient has taken the correct drug dosage. The RFID reader is embedded in the scale-top plane while the RFID tags are attached to the bottom of the bottles. In this design, when the bottle is pushed onto the plane, the reader and the tag are separated by less than one centimeter. Such close distance can ensure no interference from other tags on the platform.

Once the patient places the medicine bottle on the scale-top plane, the RFID reader reads the information of the tagged bottle and saves it into RMAIS. At the due time of taking the medicine, the system sounds an alarm and sends an SMS message to the patient's mobile phone. Meanwhile, it rotates the spoke to the correct position for the first drug to be taken. Both name and dosage of the medicine are displayed on the LCD panel. The system weighs the bottle before and after the dose is taken, to determine whether the patient is following the correct medicinal procedure.

To reduce medication errors, Sun, Wang, & Wu (2008) developed their Wisely Aware RFID Dosage (WARD) system. Because it combines barcode and RFID technologies, WARD reduces the investment expense. The order of the medications is indicated on the barcode paper printed on each medicinal unit. Furthermore, each patient is given a wristband with an attached RFID tag. The WARD system comprises a pharmacy, bedside and confirmation part.

Once a prescription is issued by the doctor, the Hospital Information System (HIS) immediately sends the order to the pharmacy. . By this way, they can begin to pack the medicine. The pharmacy part of the packing system prints the barcode label on the surface of the drug package.

These barcodes contain the information of the medicine. At the bedside, hospital staff match the information on the drug package with data stored in the RFID wristband using a PDA equipped with both barcode and RFID scanner. To ensure patient safety, the hospital staff should confirm the retrieved information with a cart monitor before the patient consumes the medication. The record is updated to the database once the medication procedure is completed.

In a pilot test, this system was highly evaluated by users. The researchers reported that the WARD system not only improves patient safety and reduces medical errors, but also lowers the investment cost to hospitals. From an ROI perspective, the integration of barcode and RFID technologies is much cheaper for hospitals than pure RFID technology.

In applying RFID technology to the healthcare industry, understanding the effects of RF electromagnetic fields on drug safety is crucially important. Bassen, Seidman, Rogul, Desta, & Wolfgang (2007) created two exposure systems that simulate the RF fields produced by real RFID systems. The aim was to study the influence of such fields on both solid and liquid medications in retail packaging and culture dishes. Culture dish is used to culture cells for various purposes. The system can expose the medicine samples at much higher levels than those 'worst-case' readers. 'worst-case' reader can only transmit and emit maximum filed strength 20cm away. Exposing the culture dishes guarantees the system to uniform electric fields and currents. In retail packaging known as container, it is used to investigate the interactions of RF with the materials. The researchers adopt two radio frequencies: ultra-high frequency (UHF) and high frequency (HF). The UHF system exposes the samples to more than 20 Watts of effective isotropic radiated power, five times the FCC (Federal Communication Commission) standards. Throughout 7 hour's measurement of the UHF system, the temperature of the medicine and the surrounding air rose by no more than 0.3 Celsius, while the field strengths were maintained within ± 0.5 dB of their expected values. Therefore, it can be concluded that RFID technology does not compromise drug safety.

2.4 Prior Implementation of RFID Blood Tracking Systems

Previous applications of RFID technology to blood tracking systems are presented in the following case studies.

Case 1: A Dynamic Blood Information Management System Based on RFID

Jiang et al (2005) proposed a RFID blood management system cooperated with fingerprint identification technology in China. This research resolved several issues of the traditional barcode approach, including the credibility of blood identification and real-time monitoring. Smart labels (RFID active tags) enabled more information to be stored and a reusable chip. The blood center and hospital could exchange the smart labels assigned to individual blood bags. Consequently, information regarding the blood and blood donor (including fingerprint images) could be accessed from the smart label and the data of the smart label eliminated afterwards.

By this approach, each blood bag is associated with unique fingerprint identification. Therefore, the possibility of impostor donation, which is absolutely illegal in China, is significantly reduced. This research is one of the earlier successful attempts to replace barcode information with a mobile RFID system to enhance security and efficiency features. However, the proposed solution was too expensive for most hospitals and blood centers in China. It did not resolve the issue of reducing the initiate setup cost for the proposed RFID system.

Case 2: Smart Blood Bag Management System in a Hospital Environment

Kim, Bae and Chang (2006) proposed a RFID-based approach for transferring blood bags from the blood bank while ensuring the quality of the blood at its ultimate delivery point. In this approach, the temperature and location information is traced in real time using RFID sensor tags. Their RFID system is based on an auto track system, which automatically delivers blood bags to their destination via rail. Because the tags are treated as a sensor network, a wide-ranging microwave frequency (2.4 GHz) is applied. In this way, the blood bags need only be packed into a container and their destination programmed.

As the container is moved, its temperatures and locations are actively tracked by the RFID system. In an experiment of the system, the blood quality was traceable and the effectiveness of delivery was improved. The live blood temperature board, location, patient and blood bag details were all monitored by the prototype software. However, active RFID devices (such as the sensor and battery) were deemed too expensive for application at that time.

Delivering blood product via rail and erecting readers along the rail exemplifies the applicability of RFID systems. The tunnel RFID system could significantly enhance the efficiency of blood delivery in hospitals and reduce the labor costs of manually delivering blood bags to each patient.

Case 3: Intelligent sensor for tracking and monitoring of blood temperature and haemoderivatives used for transfusions

Abarca et al. (2009) proposed a means of tracing and maintaining the temperature of blood and other haemoderivative products. In this research, a special sensor tag behaves as an active RFID tag that detects stores and transmits data. The tag consists of an analogic sensor and other standard components of active tags (CPU and memory). To read the data from the sensor, the tag is programmed to activate the receiver, which in turn initiates the data transmission per specified time interval.

At UHF frequencies, the detection is less disturbed by other devices and the detection range of tags within a blood bank is at least 6–7 meters. Such a short detection distance also avoids interference with other sensitive devices. Once the receiver is activated, it sends these data to a computer via USB, Ethernet or Bluetooth, so the computer updates the temperature data in real time. This RFID solution enables hospital personnel to check the temperature changes of blood bags during warming or cooling.

The proposed system can then direct other devices to halt or initiate warming or cooling by tracking the temperature in real time. However, the reliability of proposed system acquires more specific details. For instance, the operating temperature was unspecified and it may be valuable information. Nevertheless, this research is a remarkable example of cooperation between a monitor device and active tags. The customized information is recorded in the RFID chip for later access.

2.5 Safety of Blood Tracking using RFID Technology

Kozma, Speletz, Reiter, Lanzer and Wagner (2011) investigated the chemical influence of radio frequencies on blood cells over a 42-day period. The same study had been previously implemented over a shorter time period, so this study was conducted in a more practical environment. An HF (13.56 MHz) was adopted, because HF tags are the commonest type of tags applied to blood bags. Two groups of blood bags were tested. The test groups were labeled with RFID tags and data were transmitted with the tags every few seconds by the reader. Another group without radio frequency interaction was tested as the reference group.

During the experiment, the authors monitored the changes in the chemical variables, including the pH, concentrations of potassium, glucose, lactate, and hemoglobin (Hb), the hematocrit, free Hb, and hemolysis rate. The chemical variables were almost identical in the test and reference groups, implying that radio frequency technologies do not affect the integrity of blood bags. It is important that the public understand the harmlessness of radio frequencies to blood products; conveying this information would ensure public trust in the RFID system.

Similar conclusions were drawn by Hohberger, Davis, Briggs, Gutierrez and Veeramani (2012). They mentioned that radio-frequency magnetic fields usually exert no effect within the product lifetime, so no safety issues should arise from radio frequency usage in practical environments.

Chapter 3 Research Methodology

The aim of the thesis is to study a small-scale real time temperature monitoring system. Therefore, we construct a software prototype and demonstrate its usage and benefit through simulation. The research methodology must develop the software and evaluate the software outcome. Design science is the best means of defining the objective, implementing the prototype and illustrating its effectiveness. Our research is not limited to solving existing problems, but also confers new benefits or suggests an efficient idea. Thus, the design science method is used in this research to produce artifacts. Our design science approach includes research motivation, defining the objective, and system development, demonstration and evaluation. Problem identification process is excluded from this research, because the prototype aims to create a more effective solution rather than solve existing issues. Each step in the design science is introduced below.

The design science approach is preceded by a literature review, which identifies the background and current knowledge of our research field. In this research, the literature review also assists in learning the background and recent achievements of RFID technology and its application to blood tracking. A general knowledge of previous blood transfusion solutions is necessary for understanding their current state. The potential benefit can be investigated only after the previous knowledge has been learned in the literature review.

The literature review phase is followed by the design science methodology. Information system research is not restricted to the defined steps of a particular methodology, so our research requires an independent and tailored methodology model. In the Section 3.1, the design science methodology is described along with the views and opinions of previous researchers. The model of the present research is then introduced and each of its phases is explained in Section 3.2.

3.1 Design Science

It is important to introduce the design science approach and its involvement in our research. In Information Systems and related disciplines, design science research showed its success in meeting both business and research requirements (Brocke & Lippe, 2010). The design science approach has been applied for many years. DSR creates artifacts, defined as “something created by humans usually for a practical purpose” (Artifact, as cited in Geerts, 2011). As the base output of a design science research, the artifact is then used for other purposes, such as experiment and evaluation. The following is a summarized definition of DSR. Design science is “a body of theories, paradigms, models, methods and knowledge that describes and explains a proven fundamental of what design is, what happens while designing and how to improve it” (Birkhofer, 2011).

Similar to other research methodologies, design science methodology focuses on problem-solving. Design science constitutes another aspect in the information system research cycle, which creates and evaluates design science artifacts to solve identified organizational problems (Hevner, March, Park & Ram, 2004). To solve any practical problem, the solver must address two knowledge questions; first, identify the problem, and then assess whether a solution design would solve the problem. In answering the second question, the solver must predict the outcome of implementing the design in the problem domain (Wieringa, 2010). In our case, the problem is that many hospitals lack real-time temperature monitoring systems for blood transfusions. Such an automated system is predicted to assist transfusion staff to obtain the real-time temperature information of blood bags during the transfusion process.

According to Peffers, Tuunanen, Rothenberger and Chatterjee (2007), a general designs science methodology includes theory building, systems development, experimentation, and observations. However, the actual design approach does not need to adhere to this complete approach. For example, in another study conducted by Winter (2008), the DSR objectives were creating solutions to specific classes of relevant problems by a rigorous construction and evaluation process. Our own research involves four similar phases: solution proposition, system development, simulation and evaluation. By proposing an initial solution, we can decide the type of system to be developed. Once the software is developed, the system is evaluated and reviewed in a simulation study.

Many researchers deploy and assess their proposed system in a practical environment, but DSR can be carried out in many alternative ways. Besides deploying a complete functional system, the purpose of a designed artifact can be achieved by evaluating the formalized knowledge of that artifact's utility (Venable, Pries-Heje & Baskerville, 2012). In this sense, the DSR outcome can be a discussion regarding the effectiveness and other performance features of the application.

The objective of the design science approach should also match the research objectives, one of which is assessing the performance of the design artifact. This raises the question "What will be the effect of this artifact in this problem domain?" (Wieringa, 2010). In this thesis, we seek the benefits of applying temperature monitoring by RFID to blood transfusion. The design science approach can be formulated as finding the effect of the proposed application in the blood transfusion domain. Alternatively, the potential benefits might lead to new ideas and knowledge in monitoring blood temperature. Different from general design, a design science research usually produces new knowledge of interest to the community (Vaishnavi & Kuechler, 2008). Thus, the design science approach is eminently suitable as a basis for this research.

3.2 Research Model

Our research model is based on a literature review and design science methodology. The model consists of six stages as shown in Figure 3.1. The tasks of each stage are now explained in detail.

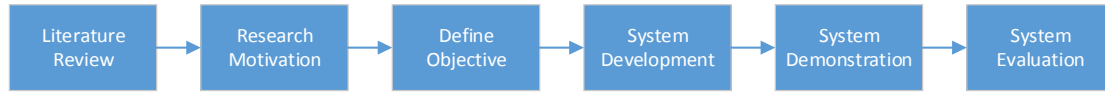


Figure 3.1 Research Model

Stage 1: Literature Review

The literature review is necessary for understanding blood tracking by RFID. The investigation focuses on RFID technology and previously proposed RFID blood tracking solutions. Regarding RFID technology, we review general RFID hardware and radio frequency standards. It should be able to review and identify the available equipment to constitute the physical components of our proposed solution.

The crux of the literature review is recent RFID blood tracking solutions as well as RFID drug tracking in general. Due to the characteristic of this thesis, the review is based on overviews rather than criticizing the past studies and pilot implementations. According to the design science approach identified in Section 3.1, we aim to identify the potentials of the proposed system rather than fixing existing issues from previous systems. Studying blood tracking systems assists in understanding how RFID technology could be applied within blood tracking systems for different purposes. Several representative solutions are selected and their operations are summarized, along with their achievements and known flaws. On the other hand, the review of drug tracking identifies the major issues in medical drug management. Because blood tracking is a part of drug tracking, these issues may also relate to blood tracking. For instance, safety issue is important in general drug tracking as well as in blood tracking.

Most of the literature was found by searching the AUT library and Google Scholar. To ensure that all literature was relevant and not outdated, the search includes items mostly less than 10 years old at the time of this research. The literature includes books and journal articles from multiple online publishers, such as ACM, IEEE and Springer.

Stage 2: Research Motivation

DSR can solve existing issues in current systems or construct solutions that are more effective and efficient than previous solutions. In Chapter 1, we identified that temperature monitoring systems are scarcely deployed. The motivation of this research is to identify and discuss a small-scale temperature monitoring system for blood transfusion. We also look for the potential features and benefits that our research solution might offer. At least, the proposed features and benefits should enable temperature monitoring and enhance (to some extent) the effectiveness and efficiency of blood transfusion process.

Stage 3: Defining Objective

At this stage, the goal of building the application is clearly identified and presented in Section 4.1. For this purpose, we require a statement of the prototype application to be built, the tasks the prototype will perform, and the benefits the prototype will achieve. Next, the infrastructure of the prototype system is described by overviewing the system. The development tools and platform are also defined at this stage. The system architecture and usage is illustrated in presentational models and workflow diagrams in Section 4.2; thus, the characteristics of the prototype application are also presented at this stage.

Stage 4: System Development

Having defined the research objective, we implement the system development phase, in which we build the prototype software. The system development proceeds through three major phases: design, development and testing. UML diagrams are used to visualize the prototype system model and describe the system behavior. This research uses case diagrams, class diagram and database diagram, all accompanied by detailed descriptions. The use case diagrams show the interactions between the system and stakeholders, including system administrators and hospital staff. Class diagram identifies the system structure, classes, methods, attributes and variables. Finally, the database diagram defines the data stored in the database of the proposed solution, and explains why this information is essential for system execution.

The graphical user interface is produced using Visual Studio and shown in Section 4.3. Web pages constructed in Visual Studio have two beneficial features. The first is quick switching between the code and design views, because any change on the code view of a web page is immediately updated on the design view. Second, a web page also has embedded C# code to directly interact with server-side codes. Server-side coding can now be accomplished by following the models and web pages with the completed graphical user interface. Essential code comments for interfaces and adapters must be provided, and must explain how future components could be adapted to the system. The database and database connection of the prototype should also be constructed. Once the prototype has been coded, a simulator function is created. The simulator serves as an uploading function of the RFID reader; specifically, it periodically uploads temperature data to the database. The simulator can be constructed either within or outside of the prototype application.

Next, a test is done for evaluating the constructed application. The test comprises unit testing; that is, each of the major functions is individually tested. The test results are recorded and described in Section 4.5.

Stage 5: System Demonstration

System demonstration interprets the prototype solution. Firstly, the prototype application is overviewed by discussing system diagrams, including use case diagrams, workflow diagrams, class diagram and database diagram. These diagrams clearly describe the architecture and behavior of the prototype.

Next, the functionalities of the system are described. The functions of the system are presented through the user interfaces, which convey the interactions between user and system. These functions are related to the described tasks in user case diagrams. The classes and methods for core functions of the application are also explained in detail by using code snippets. The core functions produce the main output of the application. Some core functions may be achieved by multiple classes and methods, so the complete workflows of the functions are also introduced.

Finally, the result of testing is discussed. Bugs or errors found during this process are highlighted and their causes are investigated in detail. Suggestions for solving the errors in future research and development are made, and elaborated in Chapter 7.

Stage 6: System Evaluation

The final stage, system evaluation, evaluates the utility and efficacy of the prototype system in assessment activities. The available resources of the proposed solution are limited in this research, chiefly comprising the prototype system and development tools. Thus, evaluation is undertaken based on a small scale simulation.

We first produce a simulation plan to instantiate the prototype for assessment. The plan is carried out by using workflow diagrams, which show the actions to be done for each simulation process. Next, the simulation results are recorded and analyzed. If any issues are found during simulation, they should be discussed as well.

Chapter 4 System Demonstration

Chapter 4 presents the developed prototype application in detail. Section 4.1 provides an overview of the proposed temperature monitoring solution describing its architecture and applied technologies for the monitoring application. Section 4.2 provides the conceptual system model representing the application, and Section 4.3 includes the screenshots of front-end user interfaces. The core functionalities of the application are explained in detail in Section 4.4. Finally, Section 4.5 tests the application and records the testing result and found issues.

4.1 Overview

As discussed in Section 1.2, there are two objectives for constructing a real-time temperature monitoring prototype. First, we must develop the fundamental tracking software that tracks the temperature data of transfused blood bags and presents the data to transfusion staff in real time. The software requirements are functionality and the resources for software development and deployment. Second, we must clarify the potential benefits of the developed tracking software. To achieve these objectives, it is necessary to build a prototype of the temperature monitoring software. Before demonstrating the prototype itself, we proposed an overall solution that monitors the temperature data of transfused blood bags.

To start with, it is important to discuss the desired hardware and its essential role in temperature monitoring of blood transfusions. If the required hardware is not existed or is unplanned, coding a temperature monitoring software becomes meaningless. In blood transfusion, blood bags are always placed within containers that maintain their required temperature over a certain period of time. The container could be outfitted with a lightweight electronic device that monitors its surface temperature during transport. In this system, RFID technology could remotely identify the blood bag and read the data on its RFID tag. RFID technology operates by reading RFID tags with a RFID reader. Many types of RFID reader are available on the market, categorized by frequency standards or usage. Remote detection of the tag attached on a blood bag is best accomplished by a handheld RFID reader. The reader connects to the internet through telecommunication services such as a GSM service. In this way, the reader can upload the tag information to a web server by launching an executable file with

upload functionality. If requested, the information details are immediately displayed to staff by an application on the web server.

Furthermore, there are specialized radio frequency sensor tags that conduct more tasks than general identification. Similar to normal RFID tags, a sensor tag is classified as active or passive. However, regardless of their status, most of these advanced tags require an internal power source to transmit the data from their sensors to the RFID reader. There are prior implementations in using RFID temperature sensor tags to detect temperature (Amador & Emond, 2010; Amin & Karmakar, 2011; Han - Pang & Chih - Peng, 2006; Kenning, 2014; Potyrailo et al., 2011; Virtanen et al., 2014). Therefore, ideally, the RFID reader and temperature sensor tag should be placed together within the transfusion container during real time monitoring tasks. The time stamp of real time event is able to be monitored. As same as most common RFID memory data, time stamps can also be stored in RFID tag (Want, 2006). Provided that the devices remain inside the container, the RFID reader can communicate with the web server and transmit the temperature data to the online database.

To view the temperature data in real time, transfusion staff should also have access to an electronic device. The data may be conveniently viewed on general web pages via web browsers, which are usually provided by operating systems installed on smart devices such as computers and smart phones. Such smart devices are reasonably affordable for hospitals. At the very least, most hospitals should have already installed workstations compatible with internal information systems. Since transfusion staff have likely used the workstation and have some fundamental experience, they should require minimal training to use the proposed application. If special requirements are identified in future, a local application client can be built based on the proposed web application.

4.1.1 Architecture

After confirming the necessary hardware for temperature monitoring, we can introduce the complete envisaged blood tracking solution, and illustrate the practical implementation of the temperature monitoring system. We also discuss the role of the prototype application in blood transfusion operations.

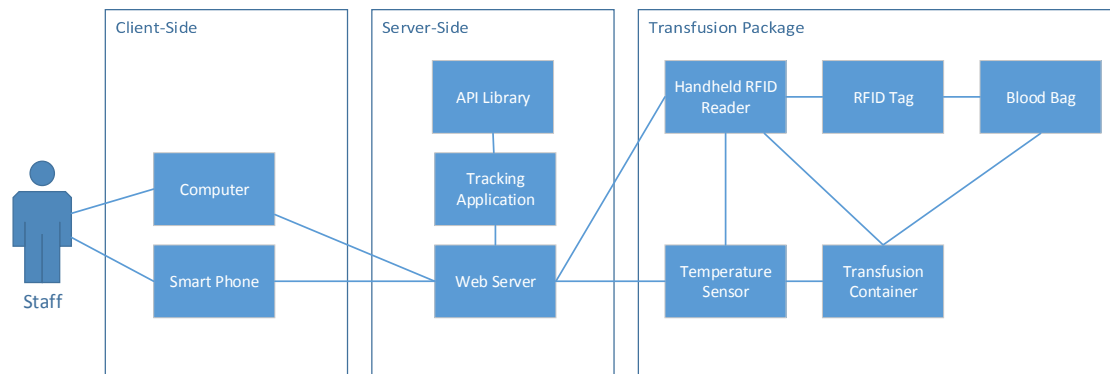


Figure 4.1 Architecture of Proposed Temperature Monitoring System

Figure 4.1 shows the necessary components and the involvement of hospital transfusion staff in the solution. First, the resources needed for the actual transfusion package are introduced. The blood bag should be attached with a regular RFID tag for identification. The bag with its fixed RFID tag is placed within a standard transfusion container and prepared for transfusion. Transfusion containers provided by North Shore Hospital have been visually confirmed as capable of carrying small handheld RFID devices. In this way, transfusion staff can manually configure the handheld reader, along with RFID sensor tag and RFID tag on blood bag. Once the reader starts communicating with the RFID tags, transfusion staff can place both the blood bag and sensor into the container and encapsulate the transfusion package. The additional preparation for blood transfusion is now completed and the temperature monitoring application can access the transfused blood bag.

The server-side part of the proposed solution contains the web server and temperature monitoring application. The application itself is detailed in the system demonstration. The application will be deployed and executed on a web server for access by RFID handheld readers and computer devices. A web server is available at all times, and hosting services for web

servers are easily available from commercial web hosting companies. It should also be noted that access to a RFID reader requires an API library file or similar resource. API library contains a group of defined functions, allowing software to access and trigger the hardware functionalities. Depending on the API library format, the coding may need to be translated for compatibility with the .NET platform on which the proposed application is based. API library is usually accessible and obtainable from the relevant RFID device manufacturer.

As previously discussed, smart devices are exemplified by computers and smart phones. Smart devices enable access to the application via web browsers and webpage displays. In this way, staff can easily communicate with the application without requiring extra devices or software, which would unnecessarily increase the investment cost. These devices are expected to be provided by the hospital organization and available to transfusion staff during the transport of blood bags. Some transfusion staff might require training to use the application, but since the application is relatively uncomplicated, tutorial related problems should be manageable by the technical department of the hospital.

4.1.2 Technologies

It is important to identify the technologies used to build the application, because they are the essential components to compose the infrastructure of the application. Several technologies were applied to build and run the prototype application, including ASP.NET, SignalR and SQL database. The version of all tools associated to selected technologies is shown in Table 4.1. Each technology would be introduced below in detail.

Tool	Version
Visual Studio	Professional 2012
ASP.NET	4
.NET	4.5.2
SignalR	2
SQL Database	Express 2012

Table 4.1 Version of Selected Tools

Visual Studio

Visual Studio is a tool enabling the development of .NET solutions, which could be conducted at a minimum of one software developer (Lair, 2009). ASP.NET web application is also one of many .NET solutions and supported by Visual Studio. Visual Studio uses an individual integrated development environment to achieve all tasks in a software development, alongside with innovative and consolidated capabilities to increase productivity (“Integrated Development Environment” n.d.). An integrated development environment known as IDE could interact with multiple languages to achieve advanced tasks. Randolph, Gardner, Anderson and Minutillo (2010) explained that IDE could satisfy developer’s need to write, compile, debug and deploy applications.

There are many advantages by using Visual Studio to build ASP.NET and other applications. Visual Studio was described as “a suite of component-based software development tools and other technologies for building powerful, high-performance applications” (“Visual Studio News” n.d.). The web development component provided by ASP.NET integrates into Visual Studio, allowing developers to conduct multiple programming tasks using the same development

platform (Johnson, 2014). Dykstra (2014) presented several new ASP.NET web development features in the recent Visual Studio 2013 version. For instance, a new feature is an upgraded user interface template that supports multiple ASP.NET frameworks (Web Forms, MVC, and Web API) (Dykstra, 2014).

ASP.NET

The prototype application is an ASP.NET Web Application. ASP.NET creates web applications based on .NET framework (Freeman, 2010). Open Source (n.d.) acknowledged that ASP.NET and .NET are open-source. Hence, we were able to legitimately build the ASP.NET application under .NET platform without requiring any further licenses. ASP.NET web forms is a component of the ASP.NET framework and Visual Studio provides tools to program ASP.NET web forms on Visual Studio IDE ("Introduction to ASP.NET," n.d.). ASP.NET web form is the user interface which would be displayed in web browser as web page. ASP.NET web pages is a single page model that usually applies C# code for server-side coding and HTML markup for client-side coding ("Get Started with ASP.NET" n.d.). HTML markup is used to draw the front end user interface and C# code is executed in the back end of web page. Mackey (2010) indicated that ASP.NET application contains a master page with a simple layout and jQuery scripts. The master page is a default background layout for multiple web pages, so the style and display of each web page becomes consistent and easy to manage.

On the other hand, the programming language of server-side coding is C#. Visual C# is "modern, high-level, multi-paradigm, general-purpose programming language for building apps using Visual Studio and the .NET Framework" ("Visual C# resources" n.d.). The advantages of C# are "simple, powerful, type-safe, and object-oriented" ("Visual C# resources" n.d.). Craig (2007) presented that C# is relatively standard comparing to many other programming languages. C# has similar conception of class and instances compared to both Java and C++ languages. There are code-behind files for every single web page to run methods on server-side. These methods can also communicate with other server-side components by using appropriate references. It means that the classes and methods are stored in C# file with '.cx' extension and the code-

behind files of web pages could invoke them. At this stage, there are only few functions involved and design pattern is not applied to reduce the complexity of prototype application.

Comet Technique

One expectation for the application is to keep uploading data to front-end dashboard without sending request for every single upload of temperature event. The real time transmission component of the application is based on Comet technique along with actual implementation from SignalR library. Thus, it is essential to introduce Comet technique and then discuss SignalR library.

Comet is a web application model that sends a long-live request to server which suspends if it reaches time-out limit or a server request is called to abandon the request (Carbou, 2011). The Comet model is different from classical polling, which repeatedly sends same request and only returns if an expected event occurs (Gravelle, n.d.). It allows the server-side functions to constantly and automatically push data to front-end web page by only establishing one long-live connection. The connection is based on a long-held request sent from front-end web page and would not terminate until user sends cancellation request or closes the active web page. By this way, the application does not require front-end web page to send request for initiating every push event.

SignalR Library

ASP.NET SignalR is a library including preset real-time communication functions, saving the extra development effort of developers from creating custom functions for the same purpose (Fletcher, 2014). SignalR was an open source project initially created by two Microsoft employees started and later adopted by Microsoft as official supported project (Mohl, 2012). The characteristic of SignalR is providing a bi-directional communication among server and multiple clients ("SignalR" n.d.). One of many SignalR features is translating client-server communication component of HTML5 Web Socket to a .NET library. SignalR applies HTML5 Web Sockets API and some parts of the following technologies - Server Sent Events, Forever Frames or Long Polling (Brind, 2012). Since SignalR library is able to implement comet technique by

integrating HTML5 Web Socket, it allows application server to consistently push data to client with a long-live request. Valdez (2012) and Taralekar (2013) pointed out several common SignalR based systems, including chat room application, real-time monitoring application, job progress update and news feed. All these applications require long-live request between client and server, so the server can instantly send update to client as needed.

After having real-time interactive communication between client and server, the real-time functionality could be added to application by calling them in SignalR hub classes. SignalR also features pre-defined C# objects and robust construction of real-time .NET application. The method and class from .NET application can be perfectly referred and invoked in SignalR hub component.

SQL Database

The storage of data is also relatively important. In our case, the real-time data stream would likely exports large amount of records and requires 24/7 access to the database server. A lot of database systems satisfy the above requirements, so we are focusing on finding a database system that could be easily integrated with prototype application under development environment. Thus, Microsoft SQL database server was selected because the integration of SQL in .NET project is mature and stabilized nowadays. Features Supported by the Editions of SQL Server 2014 (n.d.) shows that SQL server has very satisfactory general scale limit, such as computing capacity and database size. A major concern would be the maximum number of rows for temperature table, because real time data stream could easily result in large amount of records. Maximum Capacity Specifications for SQL Server (n.d.) shows that the number of rows per table is limited by available storage. In other words, theoretically, the number of rows is infinite until reaching the limit of hardware storage. This should solve the issue of saving records for real time data stream.

Availability, scalability and other performance features of SQL server are also highly satisfied and they are analyzed in detail by "Features Supported by the Editions," (n.d.). SQL express LocalDB was used to server as the database server, because it is lightweight and easy to manage overall. SQL Server Express LocalDB is a lightweight version of SQL Server Express and LocalDB

not only has the same programmable features and less prerequisites, but it also provides simple installation and easy installation process (“Editions and Components of SQL Server 2014” n.d.).

The hosting service of SQL database is also supported by most .NET web hosting service. It should not be a problem for future simulation or deployment of the application database.

SQL Dependency

The Database Engine automatically tracks dependency information when referencing entities are created, altered, or dropped and records this information in the SQL Server system catalog (“Understanding SQL Dependencies” n.d.). It requires the application to include `SqlDependency` object to reference and specify database entities. A `SqlDependency` object can be associated with a `SqlCommand` in order to detect when query results differ from those originally retrieved (“Detecting Changes with `SqlDependency`” n.d.). `SqlDependency` object would detect the latest change of specified database. It is also essential assign a delegate to the `OnChange` event, which will fire when the results change for an associated command (“Detecting Changes with `SqlDependency`” n.d.). If the database server detects changes in the application, SQL Dependency immediately triggers a method associated with the SQL Dependency object. In our case, the method detects the latest uploaded record for the proposed application.

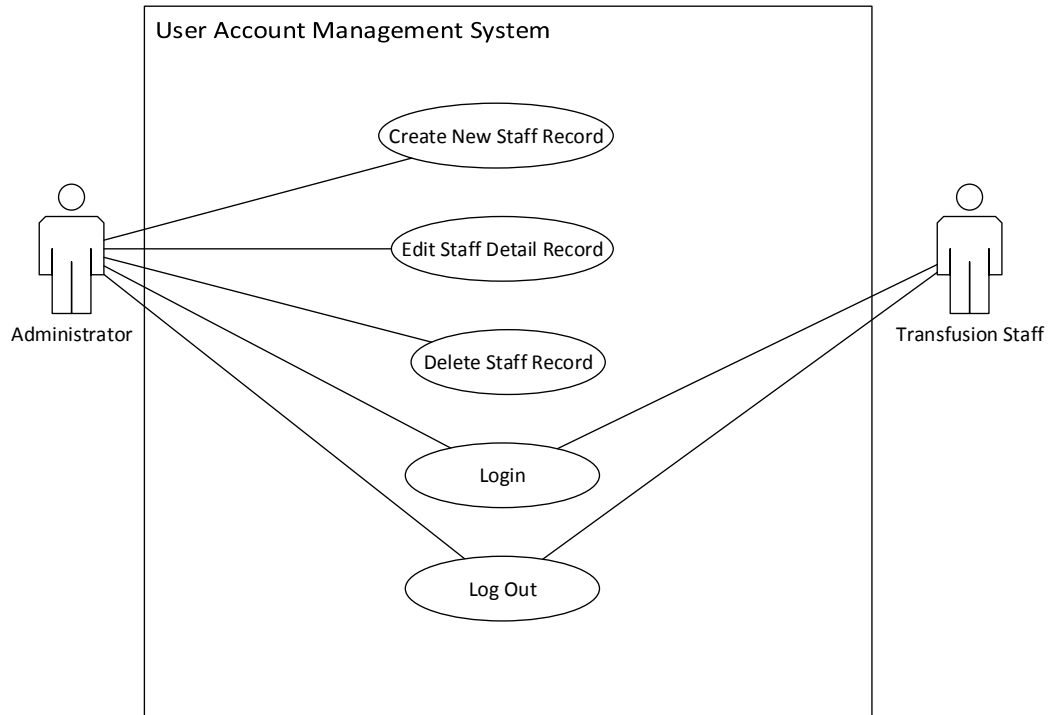
C# provides built-in object SQL Dependency to detect latest changes in SQL databases. The `SqlDependency` object represents a query notification dependency between an application and an instance of SQL Server (“`SqlDependency` Class” n.d.). The `SqlDependency` object provides multiple properties for different usage, including a way to assign a trigger method to ‘`OnChangeEventHandler`’ event handler. An application can create a `SqlDependency` object and register to receive notifications via the ‘`OnChangeEventHandler`’ event handler (“`SqlDependency` Class” n.d.).

4.2 System Model

The design of the prototype software is presented as a system model. In this project, the system models are drawn to satisfy UML standards. In Section 4.2.1, we first present the use case diagrams representing the interactions among the stakeholders and the system. It should be noted that all 'temp' keywords mean 'temperature'. Each use case represents a task that stakeholders are expected to perform on the application. The stakeholders of the present application are hospital staff and system administrators. The system administrator plays multiple important roles. First, in certain tasks requiring consent from higher level management, the administrator acts as a mediator between hospital staff and management staff. The administrator can also provide technical support during daily operation and technical assistance to hospital staff. Therefore, we construct a flowchart in Section 4.2.2 describing the several major processes in blood transfusion and their importance in temperature monitoring. The coding structure of the software is also illustrated by a class diagram shown in Section 4.2.3. Finally, the data collected in the database is clarified in a database diagram in Section 4.2.4.

4.2.1 Use Case Diagrams

User Account Management



Note: The roles of all hospital employees are unsure at the moment. At this stage, we simply indicate all staff who has right to access the system as transfusion staff.

Figure 4.2 Use Case Diagram of User Account Management System

Figure 4.2 presents the user account management of the proposed system. Account management is mandatory for security purposes, and should be fundamental at least. System access will be limited to administrators and transfusion staff, so the account details of these users must be created and updated. Since account details can be altered only by the administrator, no open-access user interface is required for creating, editing and deleting account information. A login function is required for accessing sensitive temperature and blood bag data. Staff can login with their staff ID and password, which are created with administrative assistance. Staff exiting the system should officially logout to prevent unauthorized access.

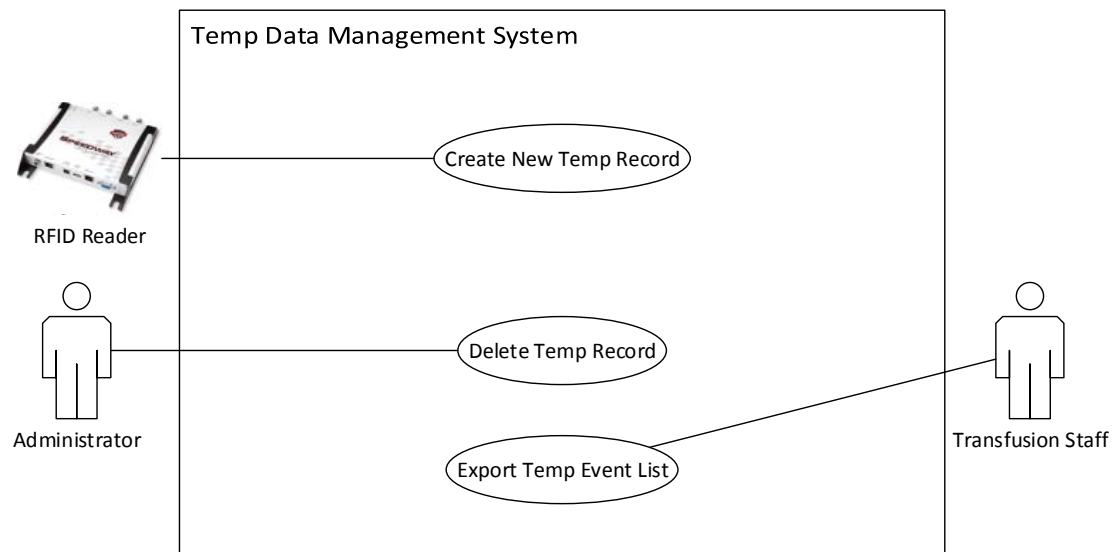
Temperature Data Management

Figure 4.3 Use Case Diagram of Temperature Data Management System

Temperature data constitute the status record of blood bags for future reference. The data are initially created by the reader and saved to an online database. For authenticity purposes, the temperature data cannot be manually updated by stakeholders, but must remain at their initial values provided by the reader. However, under exceptional circumstances they can be deleted from the database, such as when cleaning redundant data. The temperature information often supports medical treatment and research, so transfusion staff can search for a specific blood bag and export its temperature data to local files.

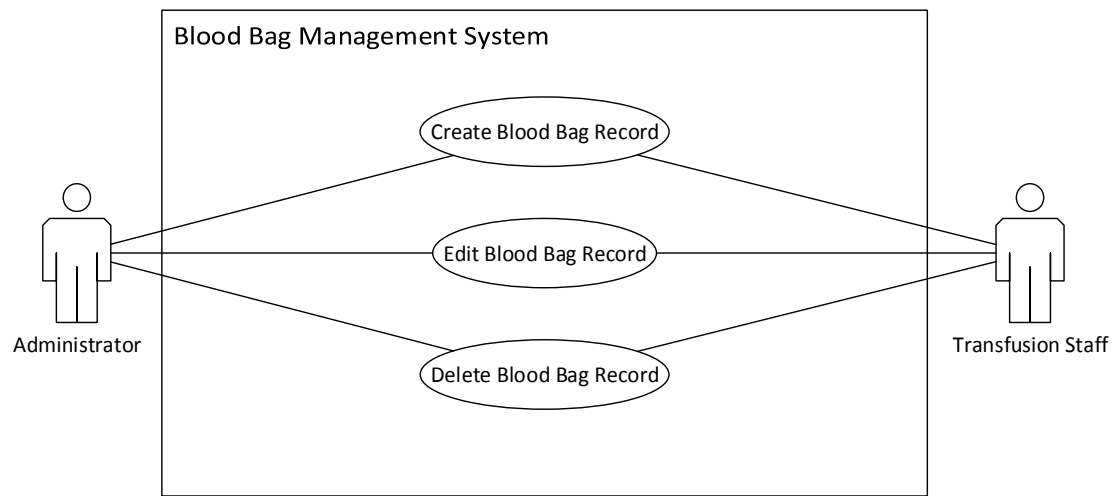
Blood Bag Management

Figure 4.4 Use Case Diagram of Blood Bag Management System

Blood bags are managed similarly to the temperature data. By accessing the information of a blood bag, transfusion staff can rapidly and simultaneously obtain both temperature and blood information. Transfusion staff and administrators can create, edit and delete blood bag records from the database. Unlike temperature data, certain attributes of the blood bags can be altered. For example, the name and description of a blood bag can be updated to maintain an accurate description as time progresses.

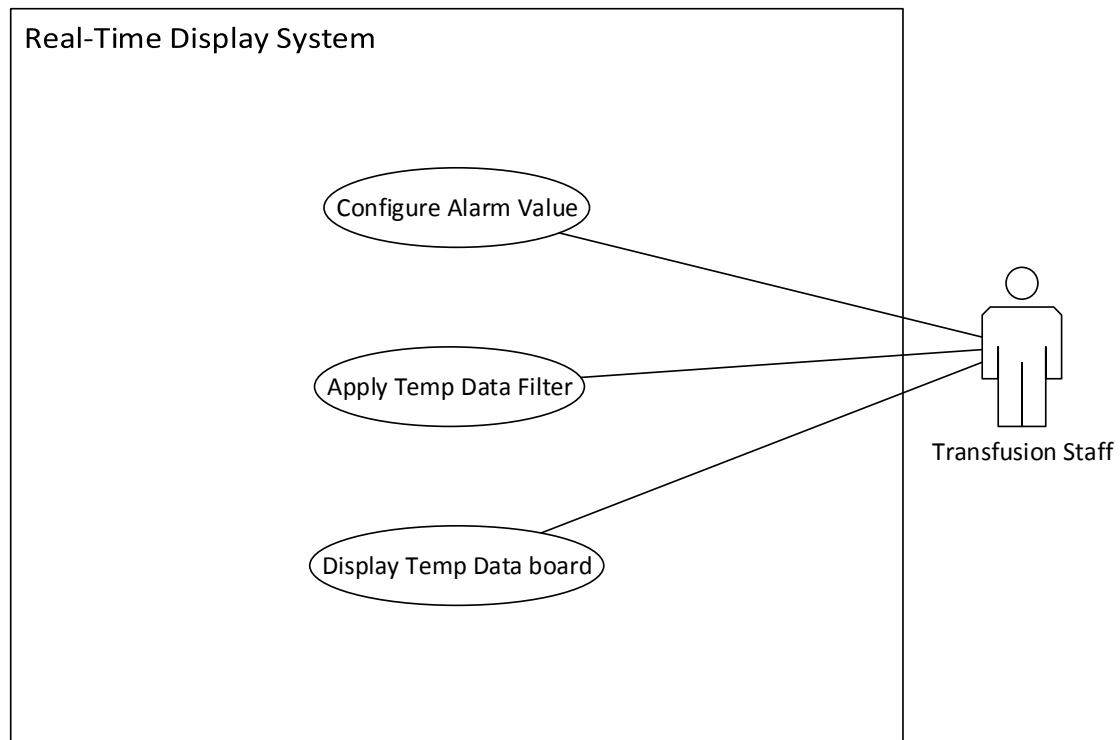
Real-Time Display Management

Figure 4.5 Use Case diagram of Real-Time Display System

Transfusion staff will use the system for online tracking of the temperature data of blood bags, which reveals the latest trend of temperature information. The real-time display component could update the latest blood bag temperature on a webpage if available. When a transfusion staff member selects a specified blood bag, the real-time board presents the latest temperature data uploaded by the RFID reader. There is also an alarm configuration that notifies transfusion staff if the blood temperature drifts outside the specified range.

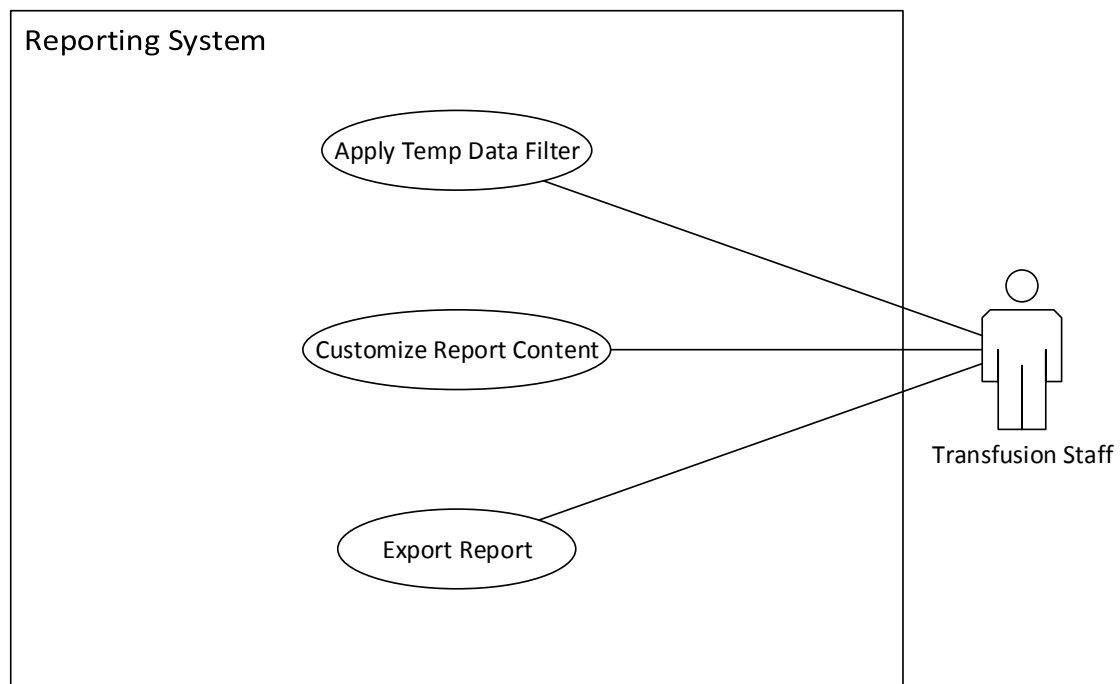
Reporting system

Figure 4.6 Use Case Diagram of Reporting System

The reporting part of the prototype produces a temperature data report showing the status of a blood bag. The temperature data report is a major output of the proposed system. It assists in identifying the status history of blood bags, as hospital requirements are more-or-less consistent. The reporting system provides functions for selecting designated temperature data and specifying certain properties of the report, such as the title. Staff can then customize their report and store it as a local file.

4.2.2 Workflow Diagram

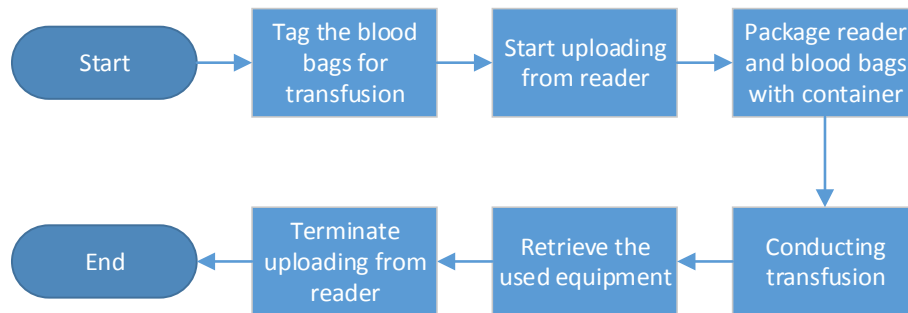


Figure 4.7 Workflow Diagram of Proposed Blood Tracking Solution

The proposed application conducts several tasks both before and after the normal transfusion process. The additional tasks related to the proposed application are outlined in Figure 4.7 above, and are describe below.

1. Transfusion staff check whether or not the blood bag is prepared with a RFID tag. If not, the staff should create a new blood bag record along with a RFID tag EPC. The corresponding RFID tag should then be attached to the blood bag.
2. Transfusion staff initiate the uploading process on the handheld RFID reader. A startup shortcut for uploading should have been prepared on the reader.
3. At the start of the uploading process, transfusion staff place the reader, temperature sensor and blood bags inside the transfusion container. The container should be properly encapsulated because the equipment may misplace during transfusion, resulting in unexpected error or inaccurate temperature recording by the temperature sensor.
4. Transfusion staff conduct a normal transfusion process.
5. At the end of the transfusion process, the transfusion equipment needs to be retrieved. The blood bag should be immediately returned to the fridge, and the RFID reader and temperature sensor detached from the container and returned to their storage locations. The handheld reader may need recharging if its battery power is low.
6. Transfusion staff manually end the uploading process on the reader and turn off the reader.

4.2.3 Class Diagram

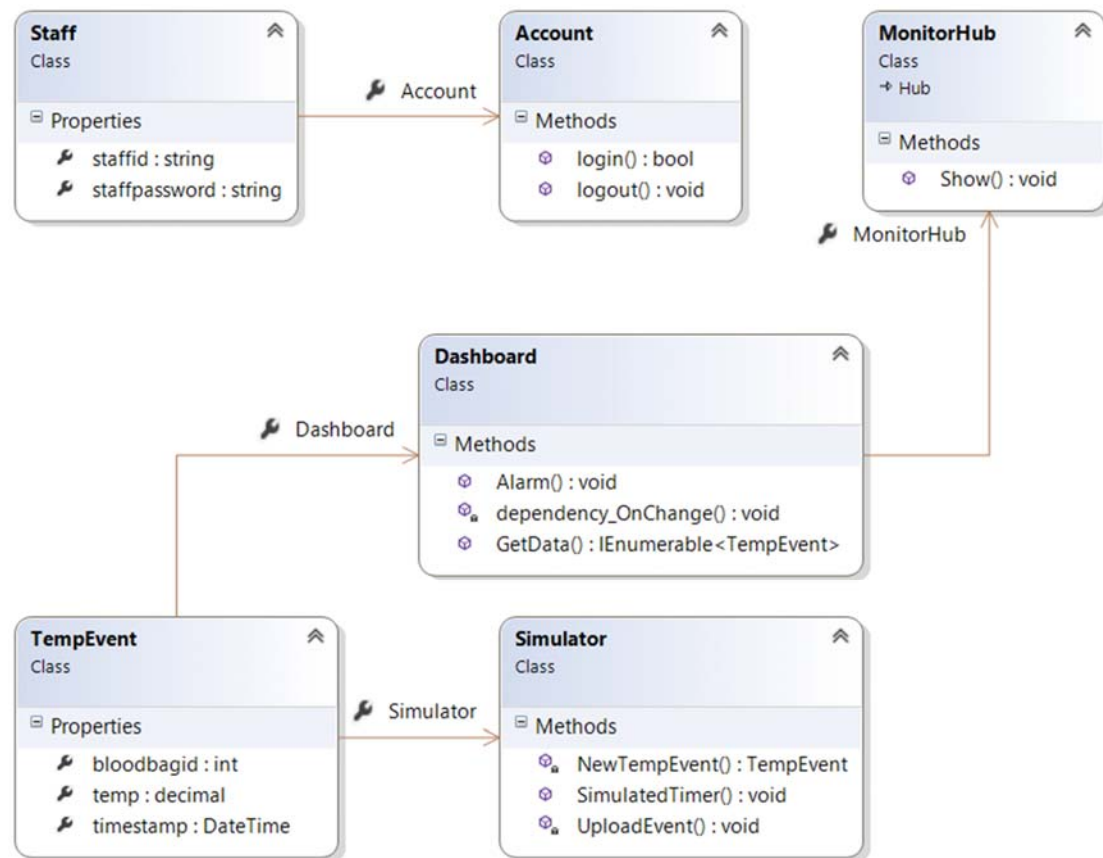


Figure 4.8 Class Diagram of Prototype Application

Figure 4.8 is a class diagram of the server-side coding structure of the prototype application. Note that this diagram excludes methods such as exporting the temperature data, which are attached as '.aspx.cs' files to the back-end file associated with the web page. Here, the extension '.aspx' refers to the web page file and '.aspx.cs' denotes the C# part of the web page coding. The methods or classes affiliated with '.aspx.cs' file are not involved in the class diagram of Figure 4.8, because they would complicate the diagram and introduce classes related to many unnecessary methods, such as button onclick method. Although button onclick is triggered when the user clicks a button, it is redundant to be discussed in the class diagram. Thus, the class diagram shows only the constructs of objects, classes including key functions, and the monitor hub class.

The account class has two methods: login and logout. The login method requires the staff id and password as inputs for validating the staff member. If the login succeeds, a new session is created with the staff id and password and the Boolean value “true” is returned. If the login fails, the Boolean value “false” is returned.

Before introducing the dashboard class, we explain the construct of a temperature event. A temperature event includes the actual temperature, a timestamp and the blood bag id, but not the event id. It is because the event id need not be displayed on the dashboard page. The dashboard class contains three methods: ‘Alarm’, ‘dependency_OnChange’ and ‘GetData’. The ‘Alarm’ method uses verification logic to compare the temperature of the blood bag with the maximum and minimum temperature limits defined by staff. The mediator method ‘dependency_OnChange’ is triggered when the application receives a SQL dependency notification. If triggered, this method calls the ‘Show’ method from SignalR hub. The ‘GetData’ method adds the ‘dependency_OnChange’ method to the SQL dependency listener and reads the latest update of the SQL database.

The ‘MonitorHub’ class implements the SignalR interface. ‘MonitorHub’ includes a ‘Show’ method that calls a sub method ‘displayTemp’, which directly invokes Javascript code functions in front-end coding. In this application, the ‘Show’ method is called by the ‘dependency_OnChange’ method triggered by SQL dependency. The ‘Show’ method then triggers the ‘displayTemp’ method, which processes Javascript codes at the front-end. The Javascript code calls the ‘GetData’ function in ‘Dashboard’ class, which reads the latest record and pastes it to the dashboard control. As a sideline, we note that one call to ‘GetData’ is invoked whenever the dashboard page is loaded, because at least one reference to ‘GetData’ is required for adding the trigger method to the SQL dependency listener.

4.2.4 Database Diagram

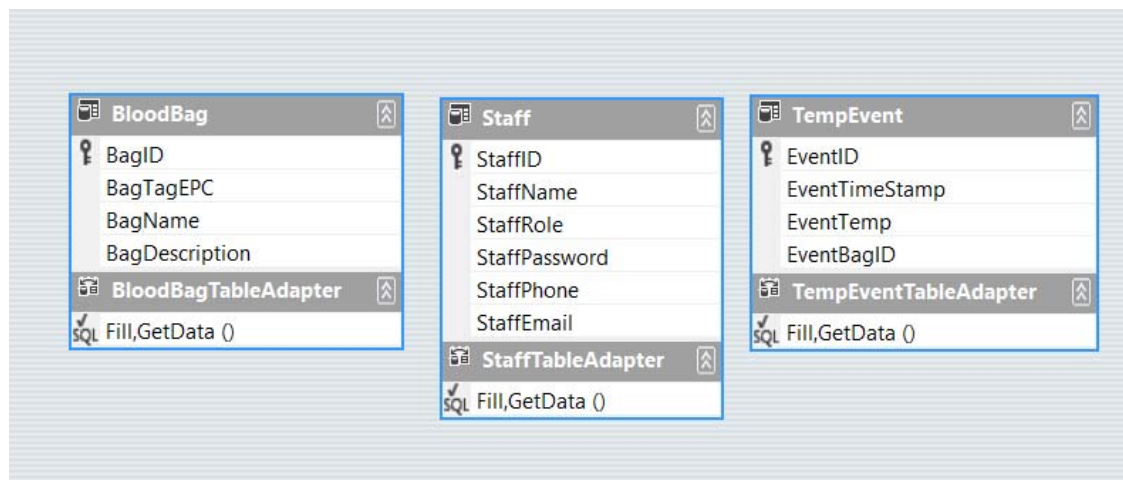


Figure 4.9 Database Diagram of Prototype Application

Figure 4.9 presents the database structure of the prototype system. Staff, temperature and blood bag data must be permanently stored in the online database. Thus, the database includes three tables for separately storing staff, temp (temperature) events and blood bags. The staff table is designed such that staff can login to the system with their authentication details (staff ID and individual password). The temperature event table records every temperature record uploaded by the RFID reader. To precisely identify the status of the blood bags, each temperature event includes the measured temperature and a timestamp accurate to one second. The blood bag table contains the fundamental information of the blood bags, including the name and general description. The description should be prepared for quick reference, because staff may require the profile of the blood bag. 'EventBagID' of the temp event table is a foreign key that matches the primary key 'BagID' of the blood bag table, because both keys are used to access information of the same blood bags for transfusion.

4.3 GUI Diagrams

The following figures are screenshots of the actual web pages captured while the prototype system was running. The data in the screenshots are not realistic, but presented for illustrative purposes only. All web pages use a unified master page. The master page is the background page of an actual web page, on which the individual web content is placed within a content place holder. In this way, the same background coding need not be added on every webpage, but can be accessed from the master page through the reference link. We now explain the tasks of each web page.

4.3.1 Home/Login Page

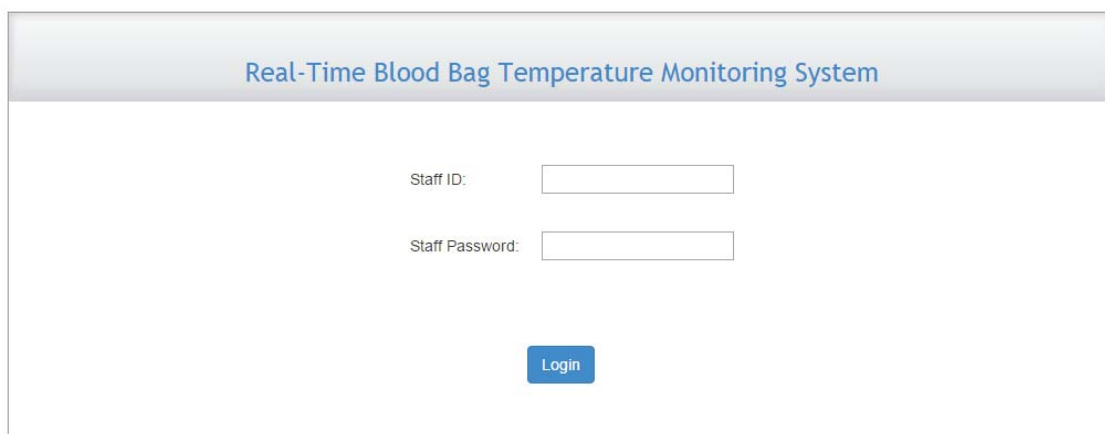
The screenshot shows a web page for the 'Real-Time Blood Bag Temperature Monitoring System'. At the top, there is a header bar with the system name in blue text. Below the header, the page has a light gray background. In the center, there are two text input fields. The first is labeled 'Staff ID:' and the second is labeled 'Staff Password:'. Below these fields is a blue button with the text 'Login' in white.

Figure 4.10 User Interface of Login Page

After entering the web address of the prototype application, the user is presented with the login page. Any staff member does not have staff id and staff password should contact the system administrator, who verifies whether the staff member is authorized to access the system. The administrator will assist the staff to create a new account and issue a staff id and password. The staff member should now be able to login by typing their staff id and staff password in the textboxes and clicking the login button below. Inappropriate login returns an error message via a dialog message box. After a successful login, a user session is created and the page automatically redirects to the main menu page. A user session is an authorized communication between the user and the server. The server always verifies the user session before staff can proceed to any other web pages of the application.

4.3.2 Main Menu Page

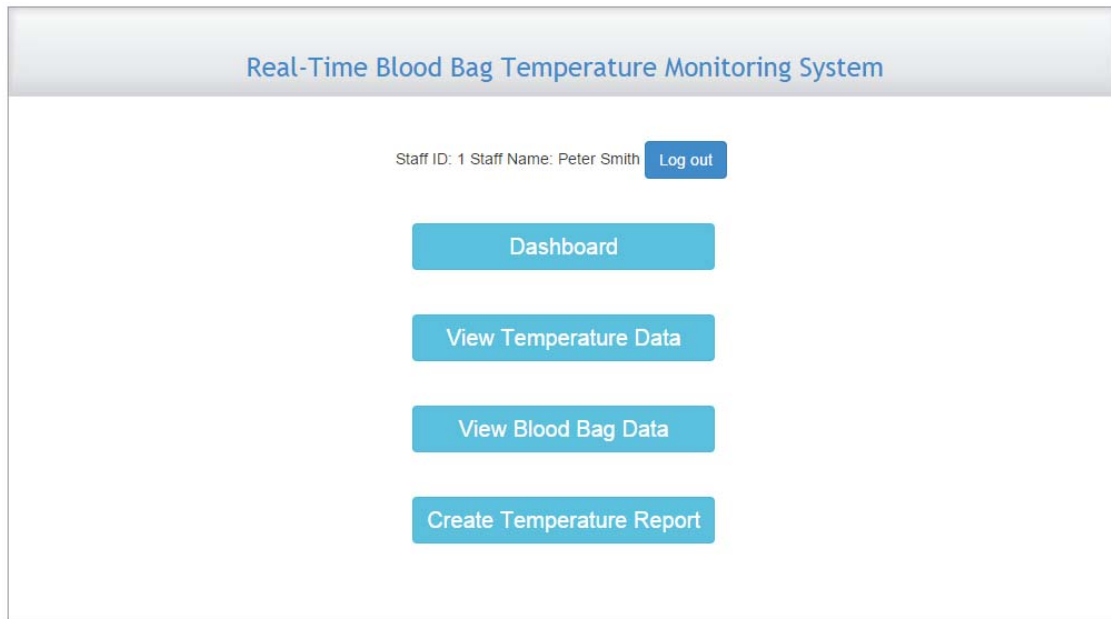


Figure 4.11 User Interface of Main Menu Page

The main menu page includes the logout button and the navigations to pages displaying the real-time temperature dashboard, temperature data, blood bag data and temperature report. Before closing the webpage, the user should logout by clicking the logout button. Many browsers do not deactivate a user session when the web page is closed. This feature allows quick restoration of closed web pages, which is useful if the user has prematurely or accidentally closed the web page. To retain this feature, we provide a log out button that staff can manually click to logout and terminate their session. If staff want to logout when visiting other pages, they can click the highlighted title at the top of the page, which returns them to the main menu page displaying the logout button.

4.3.3 Temperature Dashboard Page

Real-Time Blood Bag Temperature Monitoring System

Please Select Blood Bag ID:

Time Stamp:	14:20:20	14:20:25	14:20:30	14:20:35	14:20:40	14:20:45	14:20:60
Temperature:	6.0°C	6.2°C	6.2°C	6.5°C	6.5°C	7.0°C	7.1°C

Alarm Service

Maximum temperature:

Minimum temperature:

Figure 4.12 User Interface of Temperature Dashboard Page

The dashboard page includes the dashboard and alarm sections. In the dashboard section, staff must enter the blood bag id and click the select button to show the temperature data of the blood bag to be transfused. The temperature data may be displayed in many forms for different visual effects, such as a line graph. The preferred display might depend on the hospital requirements. For simplicity, we represent the dashboard by a table grid, and add the latest temperature data to the grid. At present the grid cannot be customized on the web page, so staff must clear the existing data in the grid by refreshing the page or re-clicking the select button. The alarm service allows staff to set an alarm when the blood bags exceed or fall below a certain temperature. The method and its logic are defined in the server-side coding. In future development, alarm actions could be customized as texts or vocal notices on the webpage, providing instant notifications to staff.

4.3.4 Temperature Data Page

Real-Time Blood Bag Temperature Monitoring System

Please enter blood bag ID to export all its temperature record:

	EventID	EventTimeStamp	EventTemp	EventBagID
Delete	2	1407209875.851	5.20	1
Delete	3	1407209876.134	5.30	1
Delete	4	1407209876.433	5.30	1
Delete	5	1407209876.717	5.40	1
Delete	6	1407209877.000	5.40	1

Figure 4.13 User Interface of Temperature Data Page

The user interface of the temperature data page displays the temperature record in a table grid. The table includes the title of each attribute and a delete button on each row. For security reasons the temperature data cannot be modified, but can be deleted to clear redundant or inappropriate records. For example, if the RFID reader starts uploading before the sensor is placed and encapsulated within the transfusion container, the sensor will record incorrect temperature details of the transfused blood bag. Such data may reasonably be deleted. Staff can also export a spreadsheet file of all temperature records of a specific blood bag. This is implemented by typing the blood bag id in the textbox and pressing the export button. If the export is successful a dialog box appears, requesting the storage location of the spreadsheet file on the local device.

4.3.5 Blood Bag Data Page

Real-Time Blood Bag Temperature Monitoring System

Create New Blood Bag Create

Tag EPC:

Name:

Description:

	BagID	Bag Tag EPC	BagName	BagDescription
Edit Delete	1	0x8988	Type ABC	
Edit Delete	3	0x8989	Type AHS	
Edit Delete	4	0x8990	Type SNM	
Edit Delete	5	0x8991	Type AKS	
Edit Delete	6	0x8992	Type ABKS	

Figure 4.14 User Interface of Blood Bag Data Page

Through the blood bag data page, staff manage the fundamental information of a blood bag. The blood bag data provides the primary information for identifying transfused blood bags. The blood bag record must be stored and updated before the blood transfusion begins, because each temperature record is uniquely associated with a blood bag id. Blood bag information may be created, updated and deleted. Therefore, this user interface features a group of controls for creating new blood bags and a table showing the existing blood bag record. Nearby the table, the staff member can edit or delete blood bag data by clicking the edit or delete buttons associated with each blood bag record. To create a new record, a user enters all the required fields in the textboxes and clicks the create button.

4.3.6 Temperature Report Page

Real-Time Blood Bag Temperature Monitoring System

Please enter blood bag ID to export temperature report:

Please enter report title:

Please enter report description:

Please enter the start time of tracking period:

< January 2015 >

Sun	Mon	Tue	Wed	Thu	Fri	Sat
28	29	30	31	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	31
1	2	3	4	5	6	7

Please enter the end time of tracking period:

< January 2015 >

Sun	Mon	Tue	Wed	Thu	Fri	Sat
28	29	30	31	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	31
1	2	3	4	5	6	7

Generate Report

Figure 4.15 User Interface of Temperature Report Page

Figure 4.15 displays the user interface for generating a temperature history report. The temperature report can be exported and downloaded in word document format with a '.docx' extension. Within this report, the temperature data of a specific blood bag can be filtered by a specific time period. The regular temperature data list is usually accompanied by designated information to be included in the report. In the prototype application, we added the report title and description as examples of customized information on the report. To generate the report, the user types the report title and report description into the relevant textboxes. These details are automatically added to the word document file on the server side. The user then selects the start and end times of the tracking period and clicks on the generate report button. A dialog box appears, requesting the storage location of the word document file on the local computer.

4.4 Key Functions of the Prototype Application

This section introduces several key functions of the application and illustrates them with code snippets. These functions constitute the most important part of the application, because they form the scaffold of the system and produce the main outputs. The complete source code of prototype application can be found in the CD attached to this thesis. Specifically, the key functions are the temperature upload function, the dashboard display function and generating temperature report function. The dashboard display function is chiefly responsible for displaying the latest temperature records on the webpage via SignalR hub and SQL Dependency. The generating temperature report function produces the demanded temperature report in word document format. However, Word is not the mandatory format for the temperature report and is adopted in this research for demonstrative purposes only. In the future, more complex converters may be added for converting the report to other formats such as Portable Document Format (PDF).

It should be noted that an actual RFID reader is not employed in this research; rather than an executable file for external FRID hardware, we implement upload methods inside the prototype application. For instance, we represent the temperature upload function by the simulator function, which performs similar tasks and achieves the same goal; namely, creating and inserting a new temperature record in the prototype application database.

4.4.1 Temperature Upload Function/Simulator function

The simulation function consists of several components. First, the object for a temperature event is defined by creating a model of the temperature event. The parameters of this object are then determined from the detected values. For simulation purpose, the temperature is randomly generated within a range of integers. The method that creates a temperature event is then called by the upload method. For simulation purposes, temperature events are updated by a timer method. The timer method periodically calls the method that creates simulated temperature event, so new temperature records are kept established and added to the database. These methods are detailed below.

```
public class TempEvent
{
    public DateTime timestamp { get; set; }
    public decimal temp { get; set; }
    public int bloodbagid { get; set; }
}
```

The class 'TempEvent' defines a single temperature event, so this specific instance will be recognized and its parameters can be modified.

```
private TempEvent NewTempEvent()
{
    // This method reads temperature value and check the timestamp
    // The temperature value is temporarily randomed between the range
    from 0 celsius to 10 celsius
    // The blood bag ID is assumed to be 5 as an example
    TempEvent newevent = new TempEvent();

    Random r = new Random();
    int i = r.Next(0, 10);

    newevent.temp = i;
    newevent.timestamp = DateTime.UtcNow;
    newevent.bloodbagid = 5;

    return newevent;
}
```

The 'NewTempEvent' method creates a new temperature event instance and sets its parameters. As mentioned in the comments within the code snippet, the simulation temperature is randomly selected between 0° C and 10° C. The timestamp is detected using the 'DateTime.UtcNow' value, known as the coordinated universal time. We also specify a proxy value, 5, as the blood bag ID. Thus, the temperature, timestamp and blood bag id become the parameters of the latest event instance.

```
private void UploadEvent(object source, ElapsedEventArgs e)
{
    TempEvent tempEvent = NewTempEvent();

    var connection = new
    SqlConnection(ConfigurationManager.ConnectionStrings["RTMS_DatabaseConnectionStrin
g"].ConnectionString);
    connection.Open();
```



```
SqlCommand cmd = new SqlCommand("INSERT INTO dbo.TempEvent  
(EventTimeStamp, EventTemp, EventBagID) VALUES (@timestamp, @temp, @bagid)",  
connection);  
cmd.Parameters.AddWithValue("@timestamp", tempEvent.timestamp);  
cmd.Parameters.AddWithValue("@temp", tempEvent.temp);  
cmd.Parameters.AddWithValue("@bagid", tempEvent.bloodbagid);  
  
cmd.ExecuteNonQuery();  
connection.Close();  
}
```

The 'UploadEvent' method calls the previous 'NewTempEvent' method to retrieve the latest detected temperature event. This method creates a new SQL query and uses the information of the latest temperature event as parameters. The insert statement of the SQL query is executed and a new row is added to the database.

```
public void SimulatedTimer()  
{  
    Timer stimer = new Timer();  
    stimer.Elapsed += new ElapsedEventHandler(UploadEvent);  
    stimer.Interval = 5000; // The reader is assumed to detect temperature  
data every 5 seconds  
    stimer.Start();  
}
```

Finally, we program a 'SimulatedTimer' method to automatically and periodically generate simulated temperature data. The timer object sets the time interval that simulates the reading interval on a RFID reader. Although the real situation is far more complex than the simulation logic, we here assume that the reader uploads at 5-second intervals. After each timer interval, a simulated temperature event is created by the 'UploadEvent' method, mimicking the upload function on a RFID reader. The above describes the basic operation of the simulated upload function.

4.4.2 Dashboard Displaying Function

The dashboard display function is implemented by components based on SignalR hub and SQL Dependency. First, the event listener of SQL Dependency is initiated on application startup and an event handler method is registered by a SQL Dependency object. In this implementation, a specified method is triggered when an application receives an event notification from SQL database. The SignalR hub is also created on the server-side. Within this hub, a method called from the server-side invokes methods from the front-end Javascript code. The front-end web page calls a proxy of SignalR hub and defines the front-end part of the hub method, enabling the Javascript part of SignalR hub to append a new temperature event to the dashboard and include additional methods. Before its record is added to the dashboard, the new event is processed through the alarm method. Since the blood bag is not displayed unless requested by the user, a logic test verifies whether the blood bag id matches the user-entered id. The program is designed to trigger the method in SignalR hub via SQL dependency. When executing, this method invoke Javascript codes that add data to the dashboard table. We first introduce the SignalR components.

```
public class MonitorHub : Hub
{
    public static void Show()
    {
        IHubContext context =
GlobalHost.ConnectionManager.GetHubContext<MonitorHub>();
        context.Clients.All.displayTemp();
    }
}
```

The 'MonitorHub' class implements the SignalR hub interface and the 'Startup' class provides additional configuration. In common with SignalR, 'MonitorHub' defines a method A called by the server-side and a method B inside method A, making it callable by the front-end web page. More specifically, the 'Show' method in 'MonitorHub' class is callable in real time on the server-side. The method 'context.Clients.All.displayTemp' is then created inside 'Show', and is triggered as 'displayTemp' whenever 'Show' is triggered. The content of 'displayTemp' is determined by Javascript code on the front-end page. Thus, the server-side can constantly send requests to the front-end to complete tasks on the user interface of the webpage in real time. In this application,

the requests are based on notifications sent from the SQL Dependency listener. The implementations of methods using SQL Dependency are explained next.

```
public class Global : System.Web.HttpApplication
{
    protected void Application_Start(object sender, EventArgs e)
    {
        SQL
        Dependency.Start(ConfigurationManager.ConnectionStrings["RTMS_DatabaseConnectionSt
ring"].ConnectionString);
    }

    protected void Application_End(object sender, EventArgs e)
    {
        SQL
        Dependency.Stop(ConfigurationManager.ConnectionStrings["RTMS_DatabaseConnectionStr
ing"].ConnectionString);
    }
}
```

The above code shows how the application initiates the SQL Dependency listener by a database connection string. The connection string connects the application to the database. The listener is terminated once the application has closed. Although this action is likely unnecessary, it prevents potential maintenance problems when the application needs to be turned off or restarted. The following method is triggered by the SQL Dependency listener.

```
private void dependency_OnChange(object sender, SqlNotificationEventArgs e)
{
    MonitorHub.Show();
}
}
```

The 'dependency_OnChange' method is triggered by a SQL Dependency notification. Its arguments are set to the standard parameters of the SQL Dependency event handler, so this method can overload the delegate provided by the SQL Dependency object. This method calls a method in the SignalR hub, thus acting as a mediator between the hub method and SQL Dependency. Although the hub method is called after the dependency notification is received,

the 'dependency_OnChange' method is still required for overloading the delegate of the SQL Dependency object as mentioned above.

```
public IEnumerable<TempEvent> GetData()
{
    using (var connection = new
SqlConnection(ConfigurationManager.ConnectionStrings["RTMS_DatabaseConnectionStrin
g"].ConnectionString))
    {
        connection.Open();
        try
        {
            using (SqlCommand command = new SqlCommand(@"SELECT
[EventID],[EventTimeStamp],[EventTemp],[EventBagID]
FROM [dbo].[TempEvent]", connection))
            {
                command.Notification = null;

                SQL_Dependency dependency = new SQL_Dependency(command);
                dependency.OnChange += new
OnChangeEventHandler(dependency_OnChange);

                if (connection.State == ConnectionState.Closed)
                    connection.Open();

                using (var reader = command.ExecuteReader())
                    return reader.Cast<IDataRecord>()
                        .Select(x => new TempEvent()
                        {
                            timestamp = x.GetDateTime(1),
                            temp = x.GetDecimal(2),
                        }).ToList();
            }
        }
        finally
        {
            connection.Close();
        }
    }
}
```

The 'GetData' method includes the core coding that reads the latest record added to the database and returns the list of temperature data. The latest record is always the last element of the list, for easy identification by other methods. 'GetData' is not directly accessible by a webpage, but is invoked by the following method, for reasons that are explained below.

```
[WebMethod]
public static Library.TempEvent NewEvent()
{
    Library.Dashboard dashboard = new Library.Dashboard();

    return dashboard.GetData().Last();
}
```

The 'GetData' method is instantiated by another method 'NewEvent' encoded in the server-side of the dashboard page. This page directly calls 'NewEvent' via a page method technique. The page method technique enables the webpage to indirectly invoke server-side codes that are not associated to the page. The web page first calls the middle method in the backend coding associated to the webpage, then incidentally calls the original method. This middle method becomes the 'NewEvent' method and is recognized by a preceding '[WebMethod]' tag. Javascript code at the front-end of the dashboard page creates a proxy representation of 'MonitorHub' and specifies the Javascript part of the code of 'displayTemp'. 'displayTemp' was previously defined in 'MonitorHub' class. The following code shows how the SignalR hub proxy is created at the front end, and how Javascript code is added to 'displayTemp', enabling actions on the user interface of the web page.

```
<script type="text/Javascript">
$(function () {
    var monitor = $.connection.monitorHub;

    // This method is declared for signalr hub to invoke Javascript code
    monitor.client.displayTemp = function () {
        getData();
    };

    $.connection.hub.start();
    getData();
});

function getData() {
    var newevent = PageMethods.NewEvent();

    // Verify the temperature by calling alarm method which includes
    custom alarm setting
    // The alarm setting includes default values for temperature variables
    PageMethods.TempAlarm(newevent.temp);

    // Check if the blood bag id is same as the input id
    if (newevent.bloodbagid.toString() ==
document.getElementById('txtBoxBloodBagID').value) {
        var row1 = document.getElementById("RowTime");
        var x = row.insertCell(0);
    }
}
```

```
        x.innerHTML = newevent.timestamp.toString();  
  
        var row2 = document.getElementById("RowTemp");  
        var y = row1.insertCell(1);  
        y.innerHTML = newevent.timestamp.toString();  
    }  
}
```

</script>

The above scripts, extracted from the dashboard page, perform several tasks. Firstly, a proxy of SignalR hub called 'MonitorHub' is created and 'displayTemp' is declared as the 'getData' method. This 'getData' method, which is unrelated to the previous 'GetData' method, calls the 'NewEvent' page method to retrieve the latest temperature data and add them to the dashboard table in the user interface. This action initiates the proxy and a confirmed single call to 'getData' is implemented immediately. The additional call to 'getData' is required because the other server-side methods cannot automatically specify the event handler method for SQL Dependency on application startup. To register the event handler method to SQL Dependency, it can be done by using the 'NewEvent' method to call the previous 'GetData' method. This is achieved by forcing a call to 'NewEvent' once the dashboard page is loaded. The 'getData' method adds the records to the table, enabling users to view the latest change without refreshing the web page.

In conclusion, the dashboard web page uses a SignalR proxy to connect the webpage to the server-side. Before the proxy starts, the child method 'displayTemp' of the 'Show' method in 'MonitorHub' is configured to invoke Javascript code at the front-end. When 'Show' is called by the SQL Dependency, it processes the 'displayTemp' method that operates the front-end user interface. The detailed working principle was explained above.

4.4.3 Alarm Function

The alarm function verifies whether the latest temperature record is within the acceptable temperature range. If the temperature is outside the desired boundary, hospital staff should be notified via alarm actions such as text messages. Although the actual alarm method has yet to be implemented in the prototype application, its verification logic has been created. The 'getData' method in the Javascript code of the dashboard page was presented above, but is redisplayed to show the role of the alarm in this application.

```
function getData() {
    var newevent = PageMethods.NewEvent();

    // Verify the temperature by calling alarm method which includes
    custom alarm setting
    // The alarm setting includes default values for temperature variables
    PageMethods.TempAlarm(newevent.temp);

    // Check if the blood bag id is same as the input id
    if (newevent.bloodbagid.toString() ==
document.getElementById('txtBoxBloodBagID').value) {
        var $tbl = $('#tblTempData');
        $tbl.append(' <tr><th>Timestamp</th><th>Tempearture</th></tr>');
        $tbl.append(' <tr><td>' + newevent.timestamp + '</td><td>' +
newevent.temp + '</td></tr>');
    }
}
```

The alarm method is also called via the page method technique, which directly calls the server-side method from the web page, and ultimately proceeds to the alarm method in the code library. The fundamental logic of the alarm method is shown below. The temperature is simply compared against the maximum and minimum temperatures specified by the user. If the temperature exceeds the maximum or falls below the minimum, an alarm action is initiated. More advanced parameters, verification logic and the actual alarm approach will be implemented in future.

```
public void Alarm(decimal maxtemp, decimal mintemp, decimal actualtemp)
{
    if(actualtemp <= mintemp)
    {
        // do something
    }

    if (actualtemp >= maxtemp)
    {
        // do something
    }
}
```

4.4.4 Generating Temperature History Report Function

To generate a temperature history report, the temperature data must be retrieved from the database and customized by a specified tracking period. First of all, the filtered data and additional information provided by the user are processed to create a new word document. This word document file is saved to the web server and then downloaded to a local machine. To preserve server storage space, the server currently holds only one temperature report per blood bag ID. The following method encodes the entire process, which is triggered by clicking the export button on the temperature report page.

```
protected void ButtonExport_Click(object sender, EventArgs e)
{
    TextBox txtBoxBagID =
    (TextBox)Page.Master.FindControl("ContentPlaceHolder1").FindControl("TextBoxBloodB
    agID");
    TextBox txtBoxTitle =
    (TextBox)Page.Master.FindControl("ContentPlaceHolder1").FindControl("TextBoxReport
    Title");
    TextBox txtBoxDescription =
    (TextBox)Page.Master.FindControl("ContentPlaceHolder1").FindControl("TextBoxReport
    Description");
    Calendar cldStartTime =
    (Calendar)Page.Master.FindControl("ContentPlaceHolder1").FindControl("CalendarStar
    tTime");
    Calendar cldEndTime =
    (Calendar)Page.Master.FindControl("ContentPlaceHolder1").FindControl("CalendarEndT
    ime");
    int i = Convert.ToInt32(txtBoxBagID.Text);
    // Read all temperature data for the specifi ID first, then filter
    with specified tracking period
    var connection = new
    SqlConnection(ConfigurationManager.ConnectionStrings["RTMS_DatabaseConnectionStrin
    g"].ConnectionString);
    SqlCommand cmd = new SqlCommand();
    SqlDataReader reader;

    cmd.CommandText = "SELECT * FROM TempEvent WHERE EventBagID="+
    txtBoxBagID.Text + ";";
    cmd.CommandType = System.Data.CommandType.Text;
    cmd.Connection = connection;

    connection.Open();
    reader = cmd.ExecuteReader();

    // Filter with start and end dates of tracking period
    List<SelectedEvent> events = new List<SelectedEvent>();
    while (reader.Read())
    {
        events = reader.Cast<System.Data.IDataRecord>()
        .Select(x => new SelectedEvent()
```



```

        {
            eventid = x.GetInt32(0),
            timestamp = x.GetDateTime(1),
            temp = x.GetDecimal(2),
            bloodbagid = x.GetInt32(3)
        }).ToList();
    }

    connection.Close();

    List<SelectedEvent> FilteredEvents = FilterEvents(events,
cldStartTime.SelectedDate, cldEndTime.SelectedDate);

    // Create the word document
    string reportfolderpath = Server.MapPath("~/TemporaryReports");
    DocX doc = DocX.Create(reportfolderpath +
"\\Temperature_History_Report_BagID_" + txtBoxBagID.Text + ".docx");
    doc.InsertParagraph(txtBoxTitle.Text + "\n");
    doc.InsertParagraph(txtBoxDescription.Text + "\n");
    doc.InsertParagraph("Blood Bag ID: " + txtBoxBagID.Text + "\n");
    string templist = "";

    foreach(SelectedEvent s in FilteredEvents)
    {
        templist += "Event ID: " + s.eventid.ToString() + "\tTime Stamp: "
+ s.timestamp.ToLongTimeString() + "\tTemperature: " + s.temp.ToString() + "\n";
    }

    doc.InsertParagraph(templist);
    doc.Save();

    // Use the custom handler to download file from server
    Response.Redirect("DownloadFile.ashx?bagid=" + txtBoxBagID.Text);
}

```

This method first retrieves the user input, which comprises the blood bag ID, report title, description, and the start and end dates of the tracking period. The temperature data list of the specified blood bag is then read from the database and filtered by the specified tracking period. Next, a word document file for temperature report with a '.docx' extension is created. A sample temperature report is shown in Appendix A. This object is based on 'Novacode' namespace, which is retrieved from Visual Studio NuGet Packages. The report title, description and each record of the filtered temperature data list are appended to the document object. The document is then saved on the server as a reference file, where it is downloadable to a user's machine by calling an asp.net handler page. The handler page comes with an '.ashx' extension, as shown in the redirect link of the above method. The handler page finds the report file on the web server and includes it in the webpage response. The user receives a general 'Save As' dialog box, requesting whether he/she wants to download the file.

4.5 Testing

The prototype was successfully compiled using Visual Studio, so the coding grammar was verified before debugging. The application was then debugged along with a virtual IIS server. While running, the entire project is filed in the server's IIS folder, and the project files are accessible via a hyperlink on the web browser. The prototype was functionally tested by several procedures. First, the administrator manually entered the blood bag information into the database via the IDE in Visual Studio. The staff account was also created directly from the database, since it cannot be registered on the web interface. Next, the login function was tested by inputting correct and unverified staff login details. During the debugging phase, the dashboard was examined by entering sample temperature data, and checking that the data were correctly updated to the web page. The data grid view and export function of the blood bag and temperature data pages were then tested. A customized report was generated from the temperature report page, which can be entered on the web page. If the page processes successfully, a dialog box requests the path of the user's machine, which determines the location of the saved temperature data report in the form of a Word document. Finally, the file is downloaded and can be opened via Microsoft Word.

Most of the components functioned as expected. However, a feature called service broker could not be added to the database via Visual Studio. This feature is essential for adding SQL notification dependency to the database. Without the service broker feature, none of the SQL Dependency related methods in the server-side coding are testable. Thus, at this stage, we cannot append the temperature data to the dashboard table. The service broker problem may be investigated and resolved in future research.

It should be also noted that the upload function is only simulated at this stage. A real upload program or feature will depend on the selected RFID reader. The upload function may significantly depend on the operating system and implementation methods; for instance, whether it should execute as a web-based background process or a local executable file stored on the reader itself. Recall that the prototype was mainly created to test its desired functionality, and was not designed to handle low-level exception. Exception handling should be carried out according to hospital requirements.

Chapter 5 System Evaluation

System evaluation assesses the artifact of design science research. In this research, the prototype application does not directly produce measurable values for the evaluation matrix. Thus, the system evaluation is based on general discussion rather than measurable evaluation criteria. Although a discussion may not provide an intuitive evaluation result, it provides subjective information and knowledge of the application by reviewing how the system executes. Thus, the evaluation proceeds by describing and analyzing the working process of the prototype application.

The discussion considers the ease of use, effectiveness, benefits and drawbacks of the prototype. Ease of use relates to users' experience of the application and their feelings when applying it. By debating the effectiveness, we can predict the performance of the application. Here, the prototype application and its effectiveness are analyzed in a simulation study. The details of the simulation process are introduced in Section 5.1. Each task of the simulation process will be analyzed and discussed in terms of the above-mentioned assessment factors. A summary is then provided in Section 5.2.

5.1 Simulation Process

The simulation process for system evaluation is conducted by debugging the prototype application on simulated temperature data created by the simulator function. Transfusion staff are assumed to have already registered with assistance from their hospital's technical support. They will undertake three workflows: preparation work, real-time operation and post operation. Preparation work includes the tasks to be completed before the transfusion commences. Real-time operations are conducted during the transfusion process, including inspecting the temperature changes on the dashboard. Post operations include the general management of database records; an important post-operational task is generating the temperature history report. Initially, each of these simulation workflows is manually conducted. The workflows are then illustrated via workflow diagrams and explanations, and their tasks are reviewed. In all simulations, the assumed user is a transfusion staff member.

5.1.1 Preparation Work

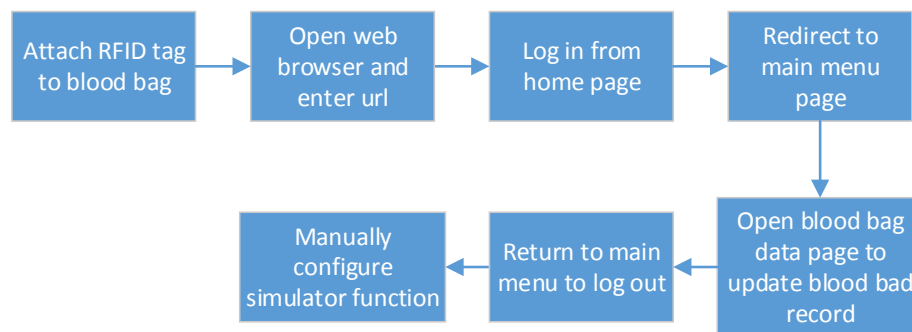


Figure 5.1 Workflow Diagram of Preparation Work

The preparation work should commence immediately after the transfused blood bag has been confirmed and prepared for placement in the transfusion container. The upload function should be started before the blood bag is actually placed in the transfusion container. The workflow includes configuring the application and initiating the upload function from the RFID reader. Because the RFID reader is currently unavailable, the upload function is unimplemented. Instead, we simulate temperature data by the simulator function and write them to the database.

Transfusion staff should first attach an RFID tag to the transfused blood bag and retrieve its EPC code. The format of the EPC code will depend on the standard of the RFID tag. Staff can then access the application from a web browser and proceed to login. Once logged in, the staff open the blood bag data page, and enter the details of the transfused blood bag along with the EPC code, or update the details of an existing blood bag in the database. After confirming that the latest information of the transfused blood bag has been deposited in the database, the staff logout from the system and complete the web page part of the preparation work. A blood bag ID should have been generated at the time of creating the blood bag record. The blood bag record is assigned an automatic identification number by the database. Finally, the simulator function is configured with the blood bag ID and executed. It should be noted that an actual upload function would require more complex configuration, since it must coordinate with the RFID reader itself and the temperature sensor.

The preparation should ideally commence in a blood bank or other blood storage place. Therefore, these places should provide workstations for staff to access the application;

otherwise, the application is accessible via their own smart devices. If there are numerous blood bag records and the user cannot directly find a record on the blood bag table, he or she may spend around one minute determining whether the blood bag record already exists in the database. To avoid this inconvenience, the application is required to search for a blood bag solely by a user-entered blood bag ID. Another identified problem is the time gap between initiating the upload function and the encapsulation of the transfusion package. Assuming that the simulation and upload functions behave in the same manner, the upload function starts slightly before the transfusion package is encapsulated. In actuality, the staff member completes the application work before placing the blood bags and RFID devices into the transfusion container. During this lapse, which might last a few seconds or minutes, the function will produce inaccurate temperature data. To solve this problem, additional features could be built into the prototype in future. For example, the upload function could delay the start time of actual execution or staff could manually start the function on the web user interface of the temperature tracking application.

5.1.2 Real-time operation

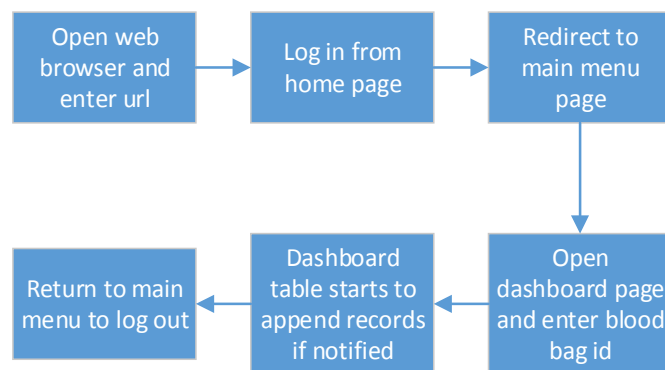


Figure 5.2 Workflow Diagram of Real-Time Operation

Real time operation is the process by which transfusion staff monitor the temperature data on the dashboard page. Tasks include accessing the application, viewing the dashboard page and leaving the application after the transfusion. Alarm tasks are not involved in this operation, because alarm notifications are not yet implemented. Alarm tasks should certainly be included if warning notifications are implemented in future development.

The workflow starts when a staff member opens the web browser and accesses the home page of the monitoring application. After a successful login, the staff member navigates to the dashboard page and enters the id number of the transfused blood bag. The latest temperature records are appended to the dashboard table on each successful entry. If the transfusion is completed or staff want to quit the system at any time, they can logout of the application by returning to the main menu page and pressing the logout button.

As mentioned in Section 4.5, the dashboard function was untestable due to technical problems with a service of the SQL database. However, the code-behind logic was carried out and the envisaged outcome was displayed in the dashboard page in Section 4.3.3. As shown in the dashboard page screenshot, each temperature update of the blood bag was expected to be added as a new column of the dashboard table. In this way, staff can directly identify the temperature status of a blood bag in real time via the web browser. However, the dashboard display presents some potential problems. Because the temperature data will probably be updated every few seconds, the table could rapidly expand beyond the screen range of the staff member's smart device. It means that the staff may experience issues in visually reading the temperature data as the actual blood transfusion goes on.

Thus, in future deployment, an alternative approach is required for displaying the temperature records. The approach is able to manage large amount of data and the control for displaying the data should be reasonable in terms of visual effect. One possible solution is to adopt AJAX extensions that obtain complex data controls, such as line graph and chart panels. AJAX extensions are the custom API library written in Javascript and XML languages, used for creating dynamic and interactive controls on webpages.

At present, the dashboard page displays the update of one blood bag per webpage. Consequently, staff must open multiple dashboard pages to view the temperature updates of multiple blood bags. This limited viewing capacity poses a major disadvantage in major hospitals conducting large-scale blood transfusions.

5.1.3 Post Operation

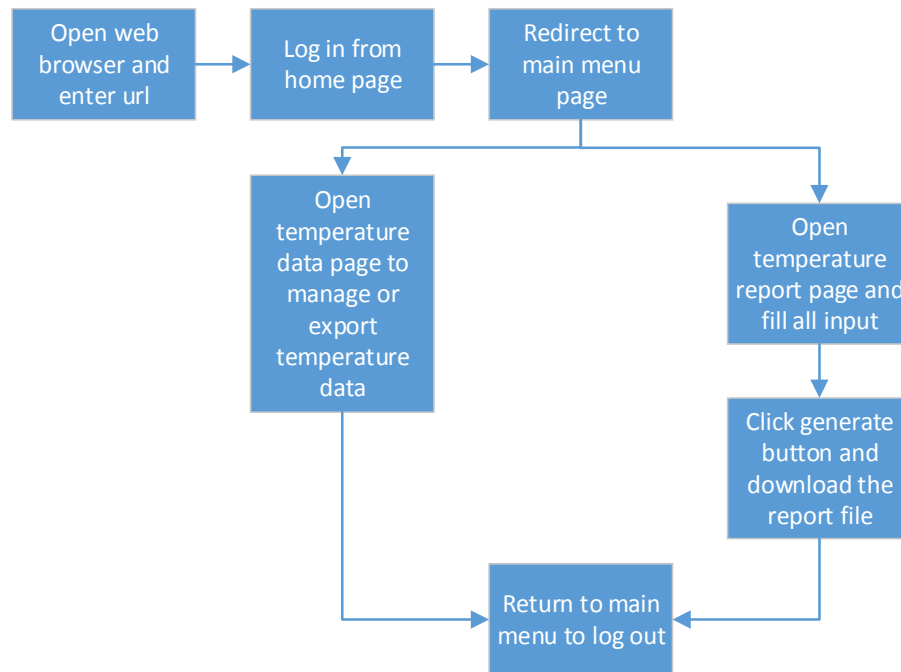


Figure 5.3 Workflow of Post Operation

Post operation involves two major tasks: accessing the temperature record via the temperature data page and generating a customized temperature history report from the temperature report page. The temperature history report differs from the spreadsheet exported from the temperature data list, because it includes a report title, a report description and other customized content.

Staff initially access the application by entering its URL address on their web browsers and proceeding to system login. An authorized staff member navigates to the temperature data page or to the temperature history page. On the temperature data page, staff can view or delete the temperature data displayed on the temperature event table, and export the temperature data to an excel file as required. They can also navigate to the temperature report page and create temperature history report. Firstly, they must insert the report title and description input, and select the start and end dates of the specified tracking period. The report file can then be downloaded from the web server. After finishing these tasks, staff should logout of the application for general security purposes.

The tasks related to temperature event records are very limited at present. Currently, the system is designed to export a data file suitable for statistical analysis. Specifically, the temperature events are exported to an 'xlsx' file which can be opened in Microsoft Excel, an application for conducting statistical analysis and generating various graphs. The content of a sample excel file is shown in Appendix B. The desired temperature event lists and descriptive contents are clearly presented in the temperature report, so they can be presented to hospital personnel not involved in transfusions. On the other hand, the customization of the temperature report page is limited. Complex customization of the report, such as text formatting and creating custom tables, requires a separate Microsoft Word application.

5.2 Summary

The prototype application was evaluated in simulation tasks based on the three workflow processes of preparation, real-time operation and post-operation, shown in Figures 5.1, 5.2 and 5.3, respectively. The simulation tasks revealed several issues and potential improvements of the application, which were identified and discussed. In the simulation of preparation work, an issue is found that the temperature sensor may produce incorrect temperature value. It is caused by a time gap between starting the upload function and encapsulating the transfusion package. During real-time operation, the dashboard display may be insufficient to show large amount of temperature events on a same table control on the web page. It is also questionable regarding the efficiency of monitoring multiple blood bags at the same time.

Chapter 6 Discussion

Section 6.1 summarizes the research and revisits the research questions raised in Chapter 1. Next, this chapter discusses the benefits and limitations of the proposed application in Sections 6.2 and 6.3, respectively.

6.1 Summary

This research investigated how web-based solutions for temperature monitoring could benefit blood transfusion processes cooperating with RFID technology. The research adhered to the research methodology of a combined literature review and design science approach. The literature review focused on the background of RFID technology, general drug tracking and prior blood transfusion systems. Next, following the design science method, a prototype application for temperature monitoring in blood transfusions was carried out. The proposed blood tracking solution and technologies used to build the monitoring application were introduced at first. The architecture and behavior of the prototype application was examined in system models. The application was then demonstrated by capturing screenshots of the user interface and code snippets of the core functions. Finally, the application was descriptively evaluated in a simulation study, assuming the envisaged transfusion workflows. The simulation revealed the potential benefits and limitations of the application.

Chapter 1 proposed three research questions, which are now answered in turn. The first question was ‘What components are required to build a web-based system for monitoring the temperature of blood bags?’ These essential components to build such a monitoring system were identified by the proposed solution in Section 4.1. The transfusion container and blood bags are already available in hospitals. We suggested that all blood bags be pasted with RFID tags with a unique EPC code. The temperature information of the tagged blood bag can then be collected by a RFID handheld reader and a RFID temperature sensor. The RFID handheld reader could be configured to communicate with both the temperature sensor and the RFID tag attached on the blood bag. Transfusion staff can start the upload function on the RFID reader and place the reader in the transfusion container, along with the temperature sensor and tagged blood bags. The temperature data can be monitored in real-time or retrieved later via a

web browser. Therefore, the application is compatible with everyday smart devices such as computers and smart phones. The temperature monitoring application needs to be prepared and installed on a web server via web hosting organizations. Regardless of how the RFID reader interacts with the application, the upload function requires the API library of RFID Reader to access and retrieve the temperature and tag data detected by the reader. Moreover, the essential technologies for constructing the monitoring application were introduced in Section 4.1.2.

The second research question was ‘What tasks could be done by blood transfusion staff once the temperature monitoring application becomes available?’ This question relates to the core functionalities provided by the temperature monitoring application. Transfusion staff can login to the application and save blood bag information in the database. Once a blood transfusion has commenced and the RFID reader begins uploading temperature data, staff can monitor the temperature of the blood bag throughout the entire transfusion phase. The application can also export a custom-designed temperature history report.

The third research question was ‘What are the potential benefits of having a temperature monitoring system in blood transfusions?’ The benefits are discussed in Section 6.2 below. The timestamp and temperature information on the dashboard page are updated in real time, so staff can identify the temperature trends of the transfused blood bag. Staff can also specify a temperature range on the dashboard page, triggering an alarm if the temperature falls outside of this range. If a blood bag overheats, staff are notified instantly and can immediately act on the blood bag. The temperature history report provides a customized timeframe or a quick overview of the blood bag’s status at a certain stage of the transfusion. An automatic export of temperature report is more efficient and faster than exporting a data storage file, followed by manual addition to a report that may require later organization.

In conclusion, this research produced a prototype of a temperature monitoring application, and assessed its potentiality and limitations in blood transfusion. The prototype application was designed and built with user interfaces and core functions. The application was tested on several envisaged scenarios in Chapter 5. After reviewing the simulation processes, the advantages and drawbacks in each simulation process were identified by descriptive analysis. Section 6.2 and Section 6.3 summarize the benefits and limitations discussed in Chapter 5.

6.2 Benefits

The temperature monitoring system satisfies several demands of monitoring the temperature data of blood bags during blood transfusions. Fundamental security of the application is paramount, and was achieved by account management. It should be noted that only transfusion staff can access the proposed system, so the information of transfusion staff can be easily referenced in the system database.

Although the present application cannot push data onto the dashboard page, we can discuss the potential benefits of this feature. If the live data were displayed on the webpage, transfusion staff could monitor the temperature information of the transported blood bags in real time. The temperature is frequently updated along with a timestamp mark, so staff could realize the temperature status of a blood bag within a known timeframe. In this way, transfusion staff could observe the temperature changes of the blood bags during the transfusion and immediately act if a problem arose.

The alarm function was designed to verify the uploaded temperatures. Staff should specify an acceptable temperature range by setting the minimum and maximum temperatures. Whether the temperature of the blood bag lies within the specified range is then determined by verification logic. Temperatures outside of the specified range will trigger the alarm notification procedure. In this way, the temperature of the transfused blood bag is actively monitored and staff are immediately alerted to a problem. Staff are not required to watch the dashboard page at all times and manually monitor the temperature changes; thus, the application greatly reduces their workload.

The application can also export data for transfusion staff. Currently, the application outputs a temperature event list and a temperature history report via a web page. Both outputs essentially show the temperature event record of blood bags, but for different purposes. The complete history of a blood bag can be derived from the temperature event list, which includes all the available temperature data for that bag.

On the other hand, the temperature data in the temperature history report can be filtered by the tracking period. Since this report can also include written descriptions, it can be used for multiple purposes. One benefit is that the temperature history report is rapidly compiled after a transfusion. The report is documented for a specified transfusion event, but presents an

efficient and convenient means of conveying temperature data to other hospital personnel under various circumstances.

Finally, this application is based on a simple ASP.NET web application. The fundamental infrastructure of the blood monitoring application was described. The architecture is feasible and uncomplicated, because the classes and methods are limited to implementing the key functionality. Only two additional features are required in the application; SignalR, to establish real-time communication between web page and server, and SQL Dependency, which triggers various methods when a new record is added to the temperature event table in the application database. Although the development tools are commercial, all coding components including the SignalR package are open source for research and development purposes.

6.3 Limitations

The limitations of the proposed application resulted from time constraints and technical limitation. These limitations are elaborated in the following paragraphs.

The most significant problem is that the dashboard cannot be tested, because the service broker is required by SQL Dependency to enable the SQL Dependency listener. Currently, enabling the SQL Dependency listener is the only approach for setting up the notification listener as a trigger mechanism when the database is updated. Without such database update notification, we cannot check how the temperature data are displayed on the dashboard page.

Another major limitation is that the upload function is simulated by generating random temperatures, which are directly added to the temperature record in the database. Implementations with actual RFID readers and temperature sensors are beyond the scope of this research. Thus, the usability and efficiency of the dashboard component of the application could not be tested in a practical blood transfusion process.

The notification approach of the temperature alarm cannot be evaluated at present. Although unacceptable temperatures were identified by the logic verification, the alarm function could not be completed without an output that would notify transfusion staff. Current resources and evidences are insufficient for deciding an appropriate alarm action.

There may be potential auditing issue if blood bag data is deleted. The purpose of enabling deleting data was removing all information of a blood bag after the bag is abandoned. However, the blood bag id in temperature event record cannot refer to this specified blood bag anymore. Although it is still possible to select temperature data with the id of deleted blood bag record, the blood bag information is lost and unable to be used to help the analysis of temperature data.

The accuracy of the uploaded temperature data in the prototype application also cannot be measured. As described in Section 4.4, the upload function currently generates random temperatures and directs them into the system database via an SQL statement. This approach assumes that the temperatures are measured to one-hundred percent accuracy, which is idealistic in practical environments. For example, transfusion staff switch on the upload function of the RFID reader before placing it in the transfusion box for encapsulation. This lag period between switch-on and placement was an identified cause of inaccurate temperature data,

because the temperature sensors and RFID readers are not instantly adjacent to the blood bags in the transfusion container. Future study would probably uncover more causes of error in the temperature data.

If many transfusion staff access the dashboard page at the same time, the limited bandwidth speed may cause delays in updating the data to the dashboard page. However, the temperature data remain correctly displayed on the dashboard control because the timestamp of the temperature event is recorded on the RFID reader. This potential problem was not investigated at this stage, because the capacity of the proposed application was unidentified. Nevertheless, it should be recognized that the number of staff and the size of the internet bandwidth are limiting variables.

Chapter 7 Future Work

Several development tasks for the prototype application could be considered in future research; implementing the upload function with a RFID reader, delaying the start and remotely terminating the upload function, advanced dashboard control, alarm implementation, enabling a service broker on the SQL database and evaluation by potential users. Each of these aspects is now discussed with regard to its importance and recommended solutions.

Implementing upload function with RFID reader

The upload function transfers the temperature data updated from the RFID reader to the server database. In the prototype application, the upload process from the RFID reader was simulated by a simulator function. However, an upload function that cooperates with the RFID reader and temperature sensor is strongly desired. If permitted by the deployment method of the upload function, the upload function can be implemented on a client executable file stored on the RFID reader. Alternatively, the temperature data transmission methods could be encoded at the server-side, and used for communication with the RFID reader and temperature sensor.

The upload function also needs to be optimized for actual use. In the scenario of the present simulator function, each upload contains one temperature event of a single blood bag. In reality, the transfusion container may contain several blood bags, meaning that multiple RFID tags are involved. In this case, the RFID reader must be able to simultaneously read and upload multiple temperature datasets.

Delaying the start and remotely terminating of the upload function

In Section 5.1.1, we found that inaccurate temperature data will be added to the database during actual blood transfusion operations. This problem may be mitigated by delaying the start time of the upload function. This may be achieved by delaying the execution of reading and uploading the temperature data. A default delay time could be defined as a constant in the source code of the upload function. Alternatively, the delay time could be manually specified in a front-end user interface of the upload function, created for that purpose.

The upload function could be remotely terminated on its own web page. The development of remote termination is optional and depends on the specific requirements. For instance, the upload function could be terminated once the transfused blood bag is emptied. It should be noted that, to prevent inaccurate temperature data when retrieving the RFID reader from the transfusion container, the RFID reader can be placed on top of the blood bags and the temperature sensor in the transfusion container. For example, staff can terminate the upload function by simply removing the cover of the transfusion container and directly accessing the front of the RFID handheld reader.

Advanced dashboard control

Table control proved insufficient for handling and displaying the temperature data of a transfused blood bag. The temperature record is usually updated every few seconds by the RFID reader and might be accumulated over several hours during blood tracking. Therefore, a general table control would easily exceed the boundary of the browser. Thus, we must find an alternative means of managing and displaying long-duration temperature data on the dashboard web page. We recommend complex chart control, which displays the temperature data on a two-dimensional graph. Active chart control can be implemented on web pages in several ways. For example, in asp.net applications, chart control is often implemented by an AJAX technique. For this implementation, AJAX may be used for active data transmission and records are added to the chart control by functions in the Javascript part of the code library of the AJAX extension.

Alarm implementation

This research developed the interface of the alarm function and identified the parameters of alarm setting. Staff can manually configure the maximum and minimum temperatures. If the measured temperature falls outside of the specified range, the alarm function is triggered and notification actions are taken. In the current application, the alarm settings are processed by a handler method on the dashboard page, but the alarm notification action was not implemented. Due to the time limitations of this research an alarm action for efficient real-time notification

was not identified. An in-depth investigation of transfusion staffs' preferred alarm notification and the achievable notification methods require investigation in future study.

Enabling service broker on SQL database

The prototype application actively triggers methods using SQL notification dependency whenever a new record is added to the database. SQL notification dependency is a query notification dependency, and requires enabling of a service broker on the SQL database server. SQL Server Service Broker supports messaging and queuing applications running under the SQL Server Database Engine ("SQL Server Service Broker" n.d.). In this research, the entire dashboard component was unusable, because the service broker could not be enabled by SQL query in Visual Studio. To resolve this problem, the database of the prototype application could be setup in SQL management studio for advanced investigation in future development. SQL management studio provides complicated configuration and management functionality for SQL databases. It should contain sufficient diagnostic tools to identify the errors preventing addition of the service broker feature.

Evaluation by potential users

The prototype was preliminarily assessed by the researcher in Chapter 5. However, the researcher has limited knowledge of blood tracking and the evaluation result is narrow. In order to gain the opinions from experts in this field, the application needs to be tested by healthcare organizations in the future. For example, blood donation centre, district health board and local hospital are the possible organizations that might employ and test the application with their current blood transfusion systems. They have undergoing blood transfusion process and experienced transfusion staff, who will likely provide professional feedbacks and suggestions.

Chapter 8 References

- Abarca, A., de la Fuente, M., Abril, J. M., García, A., & Pérez-Ocón, F. (2009). Intelligent sensor for tracking and monitoring of blood temperature and hemoderivatives used for transfusions. *Sensors and Actuators A: Physical*, 152(2), 241-247. doi: <http://dx.doi.org/10.1016/j.sna.2009.03.018>
- Ajami, S., & Rajabzadeh, A. (2013). Radio frequency identification (RFID) technology and patient safety. *Journal of Research in Medical Sciences*, 18(9), 809-813. Retrieved from <http://ezproxy.aut.ac.nz/login?url=http://search.proquest.com/docview/1467825724?accountid=8440>
- Amador, C., & Emond, J. P. (2010). Evaluation of sensor readability and thermal relevance for RFID temperature tracking. *Computers and Electronics in Agriculture*, 73(1), 84-90. doi: <http://dx.doi.org/10.1016/j.compag.2010.04.006>
- Amin, E. M., & Karmakar, N. (2011, 28-31 Oct. 2011). Development of a chipless RFID temperature sensor using cascaded spiral resonators. Paper presented at the Sensors, 2011 IEEE.
- Bang, J.-H., Chinzorig, B.-O., Koh, H.-S., Cha, E.-J., & Ahn, B.-C. (2012). A Small and Lightweight Antenna for Handheld RFID Reader Applications. *Antennas and Wireless Propagation Letters*, IEEE, 11, 1076-1079. doi: 10.1109/LAWP.2012.2217311
- Barton, R. J., Kennedy, T. F., Williams, R. M., Fink, P. W., Ngo, P. H., & Ingle, R. R. (2010, 6-13 March 2010). Detection, identification, location, and remote sensing using SAW RFID sensor tags. Paper presented at the Aerospace Conference, 2010 IEEE.
- Birkhofer, H. (2011). From design practice to design science: the evolution of a career in design methodology research. *Journal of Engineering Design*, 22(5), 333-359. doi: 10.1080/09544828.2011.555392
- Blood transfusion; tracking donated blood with radio-frequency labels. (2011). *Politics & Government Business*, , 52. Retrieved March 15, 2015, from <http://ezproxy.aut.ac.nz/login?url=http://search.proquest.com/docview/907074660?accountid=8440>

- Bocchi, L., & Di Giacomo, P. (2007). System for Tracing of blood transfusions and RFID. In T. Jarm, P. Kramar & A. Zupanic (Eds.), 11th Mediterranean Conference on Medical and Biomedical Engineering and Computing 2007 (Vol. 16, pp. 1062-1065): Springer Berlin Heidelberg.
- Brind, M. (2012, December 27). SignalR And Knockout In ASP.NET Web Pages Using WebMatrix. Retrieved March 17, 2015, from <http://www.mikesdotnetting.com/article/206/signalr-and-knockout-in-asp-net-web-pages-using-webmatrix>
- Carr, A. S., Zhang, M., Klopping, I., & Min, H. (2010). RFID Technology: Implications for Healthcare Organizations. American Journal of Business, 25(2), 25-40. doi: doi:10.1108/19355181201000008
- Carbou, M. (2011, July 19). Reverse Ajax, Part 1: Introduction to Comet. Retrieved March 10, 2015, from <http://www.ibm.com/developerworks/library/wa-reverseajax1/>
- Chainway C3000. (n.d.). Retrieved March 16, 2015, from <http://www.chainway.net/products/C3000.asp>
- Charatan, F. (1999, December 11). Medical errors kill almost 100000 Americans a year. Retrieved March 17, 2015, from <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1117251/>
- Chen, W.-S., & Huang, Y.-C. (2013). A Novel CP Antenna for UHF RFID Handheld Reader. Antennas and Propagation Magazine, IEEE, 55(4), 128-137. doi: 10.1109/MAP.2013.6645148
- Craig, I. (2007). C# Object-Oriented Programming Languages: Interpretation (pp. 201-230): Springer London.
- Davis, R., Geiger, B., Gutierrez, A., Heaser, J., & Veeramani, D. (2009). Tracking blood products in blood centres using radio frequency identification: a comprehensive assessment. Vox Sanguinis, 97(1), 50-60. doi: 10.1111/j.1423-0410.2009.01174.x
- de Grandmont, M. J., Ducas, É., Girard, M., Méthot, M., Brien, M., & Thibault, L. (2014). Quality and safety of red blood cells stored in two additive solutions subjected to multiple room temperature exposures. Vox Sanguinis, 107(3), 239-246. doi: 10.1111/vox.12154

- Detecting Changes with SqlDependency. (n.d.). Retrieved March 17, 2015, from [https://msdn.microsoft.com/en-us/library/62xk7953\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/62xk7953(v=vs.110).aspx)
- Dykstra, T. (2014, December 1). Creating ASP.NET Web Projects in Visual Studio 2013. Retrieved March 16, 2015, from <http://www.asp.net/visual-studio/overview/2013/creating-web-projects-in-visual-studio>
- Editions and Components of SQL Server 2014. (n.d.). Retrieved March 17, 2015, from <https://msdn.microsoft.com/en-us/library/ms144275.aspx>
- Electronic Product Code / Radio Frequency Identification (RFID). (n.d.). Retrieved March 14, 2015, from <http://www.gs1.org/epc-rfid>
- EPC/RFID Certification. (n.d.). Retrieved March 14, 2015, from <http://www.gs1.org/epcrfid-certification>
- Fachberger, R., Binder, A., & Erlacher, A. (2009, 25-28 Oct. 2009). SAW-RFID and temperature monitoring of slide gate plates. Paper presented at the Sensors, 2009 IEEE.
- FDA clears first blood tracking device that uses radio frequency identification technology. (2013, May 28). *Targeted News Service*. Retrieved March 15, 2015, from <http://ezproxy.aut.ac.nz/login?url=http://search.proquest.com/docview/1355865144?accountid=8440>
- Features Supported by the Editions of SQL Server 2014. (n.d.). Retrieved March 11, 2015, from <https://msdn.microsoft.com/library/cc645993.aspx>
- Finkenzeller, K., & Müller, D. (2010). RFID Handbook : Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication Retrieved from <http://AUT.eblib.com.au/patron/FullRecord.aspx?p=547191>
- Fletcher, P. (2014, June 10). Introduction to SignalR. Retrieved March 15, 2015, from <http://www.asp.net/signalr/overview/getting-started/introduction-to-signalr>
- Freeman, A. (2010). ASP.NET Introducing Visual C# 2010 (pp. 1099-1124): Apress.
- Gallagher, M. W., & Malocha, D. C. (2013). Mixed orthogonal frequency coded SAW RFID tags. Ultrasonics, Ferroelectrics, and Frequency Control, IEEE Transactions on, 60(3), 596-602. doi: 10.1109/TUFFC.2013.2601

- Geerts, G. L. (2011). A design science research methodology and its application to accounting information systems research. *International Journal of Accounting Information Systems*, 12(2), 142-151. doi: <http://dx.doi.org/10.1016/j.accinf.2011.02.004>
- Get Started with ASP.NET. (n.d.). Retrieved March 16, 2015, from <http://www.asp.net/get-started>
- Girbau, D., Ramos, A., Lazaro, A., Rima, S., & Villarino, R. (2012). Passive Wireless Temperature Sensor Based on Time-Coded UWB Chipless RFID Tags. *Microwave Theory and Techniques, IEEE Transactions on*, 60(11), 3623-3632. doi: 10.1109/TMTT.2012.2213838
- Glover, B., Bhatta, H. (2006). *RFID Essentials*. Sebastopol, CA: O'Reilly Media.
- Gravelle, R. (n.d.). Comet Programming: The Hidden IFrame Technique. Retrieved March 11, 2015, from <http://www.webreference.com/programming/Javascript/rg30/index.html>
- Gupta, G., Singh, B. P., Bal, A., Kedia, D., & Harish, A. R. (2014). Orientation Detection Using Passive UHF RFID Technology [Education Column]. *Antennas and Propagation Magazine, IEEE*, 56(6), 221-237. doi: 10.1109/MAP.2014.7011063
- Gutierrez, A., Levitt, J., Reifert, D., Raife, T., Diol, B., Davis, R., & Veeramani, R. (2013). Tracking blood products in hospitals using radio frequency identification: Lessons from a pilot implementation. *ISBT Science Series*, 8(1), 65-69. doi: 10.1111/voxs.12015
- Han-Pang, H., & Chih-Peng, L. (2006). Design of combined voltage reference and temperature sensor for RFID applications. *Sensor Review*, 26(2), 106-107. doi: 10.1108/02602280610652686
- Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly*, 28(1), 75-105. doi: 10.2307/25148625
- Hohberger, C., Davis, R., Briggs, L., Gutierrez, A., & Veeramani, D. (2012). Applying radio-frequency identification (RFID) technology in transfusion medicine. *Biologicals*, 40(3), 209-213. doi: <http://dx.doi.org/10.1016/j.biologicals.2011.10.008>
- Huang, T.-J., & Hsu, H.-T. (2012, 8-14 July 2012). Log-periodic dipole array with improved front-to-back ratio for universal UHF RFID handheld reader applications. Paper presented at the Antennas and Propagation Society International Symposium (APSURSI), 2012 IEEE.

- Integrated Development Environment. (n.d.). Retrieved March 16, 2015, from <https://www.visualstudio.com/features/ide-vs>
- Introduction to ASP.NET Web Forms. (n.d.). Retrieved March 16, 2015, from <http://www.asp.net/web-forms/what-is-web-forms>
- Johnson, B. (2014). Professional Visual Studio 2013 Retrieved from <http://AUT.ebib.com.au/patron/FullRecord.aspx?p=1642646>
- Jun, Y., Jun, Y., Law, M. K., Yunxiao, L., Man Chiu, L., Kwok Ping, N., . . . Yuen, M. (2010). A System-on-Chip EPC Gen-2 Passive UHF RFID Tag With Embedded Temperature Sensor. *Solid-State Circuits, IEEE Journal of*, 45(11), 2404-2420. doi: 10.1109/JSSC.2010.2072631
- Kabachinski, J. (2005). An introduction to RFID. *Biomedical Instrumentation & Technology*, 39(2), 131-134. Retrieved March 15, 2015, from <http://ezproxy.aut.ac.nz/login?url=http://search.proquest.com/docview/208685686?accountid=8440>
- Kang, A., Zhang, C., Ji, X., Han, T., Li, R., & Li, X. (2013). SAW-RFID enabled temperature sensor. *Sensors and Actuators A: Physical*, 201(0), 105-113. doi: <http://dx.doi.org/10.1016/j.sna.2013.06.016>
- Kenning, G. G. (2014, 8-10 April 2014). Development of a nanoparticle time-temperature sensor for passive and active RFID. Paper presented at the RFID (IEEE RFID), 2014 IEEE International Conference on.
- Kim, S.-J., Yoo, S., Kim, H.-O., Bae, H.-S., Park, J.-J., Seo, K.-J., & Chang, B.-C. (2006). Smart Blood Bag Management System in a Hospital Environment. In P. Cuenca & L. Orozco-Barbosa (Eds.), *Personal Wireless Communications* (Vol. 4217, pp. 506-517): Springer Berlin Heidelberg.
- Ko, C.-H. (2009). RFID-based building maintenance system. *Automation in Construction*, 18(3), 275-284. doi: <http://dx.doi.org/10.1016/j.autcon.2008.09.001>
- Kocer, F., & Flynn, M. P. (2006). An RF-powered, wireless CMOS temperature sensor. *Sensors Journal, IEEE*, 6(3), 557-564. doi: 10.1109/JSEN.2006.874457

- Kozlovski, N. Y., Malocha, D. C., & Weeks, A. R. (2011). A 915 MHz SAW Sensor Correlator System. *Sensors Journal, IEEE*, 11(12), 3426-3432. doi: 10.1109/JSEN.2011.2159856
- Kozma, N., Speletz, H., Reiter, U., Lanzer, G., & Wagner, T. (2011). Impact of 13.56-MHz radiofrequency identification systems on the quality of stored red blood cells. *Transfusion*, 51(11), 2384-2390. doi: 10.1111/j.1537-2995.2011.03169.x
- Lair, R. (2009). *Introduction to Visual Studio 2008 Beginning Silverlight 3* (pp. 13-37): Apress.
- Landt, J. (2005). The history of RFID. *Potentials, IEEE*, 24(4), 8-11. doi: 10.1109/MP.2005.1549751
- Law, M. K., Bermak, A., & Luong, H. C. (2010). A Sub- μ W Embedded CMOS Temperature Sensor for RFID Food Monitoring Application. *Solid-State Circuits, IEEE Journal of*, 45(6), 1246-1255. doi: 10.1109/JSSC.2010.2047456
- Lin, Y.-F., Lee, C.-H., Pan, S.-C., & Chen, H.-M. (2013). Proximity-Fed Circularly Polarized Slotted Patch Antenna for RFID Handheld Reader. *Antennas and Propagation, IEEE Transactions on*, 61(10), 5283-5286. doi: 10.1109/TAP.2013.2272677
- Lorenzo, J., Girbau, D., Lázaro, A., & Villarino, R. (2014). Temperature sensor based on frequency-coded chipless RFID tags. *Microwave and Optical Technology Letters*, 56(10), 2411-2415. doi: 10.1002/mop.28599
- Mackey, A. (2010). *ASP.NET Introducing .NET 4.0* (pp. 225-249): Apress.
- Martinez Brito, J. P., & Rabaeijs, A. (2013, 2-6 Sept. 2013). CMOS smart temperature sensors for RFID applications. Paper presented at the Integrated Circuits and Systems Design (SBCCI), 2013 26th Symposium on.
- Maximum Capacity Specifications for SQL Server. (n.d.). Retrieved March 15, 2015, from <https://msdn.microsoft.com/en-us/library/ms143432.aspx>
- Miesen, R., Kirsch, F., & Vossiek, M. (2013). UHF RFID Localization Based on Synthetic Apertures. *Automation Science and Engineering, IEEE Transactions on*, 10(3), 807-815. doi: 10.1109/TASE.2012.2224656
- Ming, J., Ping, F., Hexin, C., Mianshu, C., Bo, X., Zhonghua, S., . . . Yu, W. (2005, 17-18 Jan. 2006). A Dynamic Blood Information Management System Based on RFID. Paper presented at

- the Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the.
- Mohamad, S., Fang, T., Amira, A., Bermak, A., & Benammar, M. (2013, 26-28 Aug. 2013). A low power oscillator based temperature sensor for RFID applications. Paper presented at the Quality Electronic Design (ASQED), 2013 5th Asia Symposium on.
- Mohl, D. (2012). Building Web, Cloud, and Mobile Solutions with F#: O'Reilly.
- Opasjumsruskit, K., Thanthipwan, T., Sathusen, O., Sirinamarattana, P., Gadmanee, P., Pootarapan, E., . . . Thamsirianunt, M. (2006). Self-powered wireless temperature sensors exploit RFID technology. *Pervasive Computing, IEEE*, 5(1), 54-61. doi: 10.1109/MPRV.2006.15
- Open Source. (n.d.). Retrieved March 16, 2015, from <http://www.asp.net/open-source>
- Peffer, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 24(3), 45-77. doi: 10.2307/40398896
- Plessky, V. P., & Reindl, L. M. (2010). Review on SAW RFID tags. *Ultrasonics, Ferroelectrics, and Frequency Control, IEEE Transactions on*, 57(3), 654-668. doi: 10.1109/TUFFC.2010.1462
- Potyrailo, R. A., Wortley, T., Surman, C., Monk, D., Morris, W. G., Vincent, M., . . . Ehring, H. (2011). Passive multivariable temperature and conductivity RFID sensors for single-use biopharmaceutical manufacturing components. *Biotechnology Progress*, 27(3), 875-884. doi: 10.1002/btpr.586
- Randolph, N., Gardner, D., Anderson, C., & Minutillo, M. (2010). Professional Visual Studio 2010 Retrieved from <http://AUT.eblib.com.au/patron/FullRecord.aspx?p=547042>
- Reindl, L. M., Pohl, A., Scholl, G., & Weigel, R. (2001). SAW-based radio sensor systems. *Sensors Journal, IEEE*, 1(1), 69-78. doi: 10.1109/JSEN.2001.923589
- RFID advance to improve safety of nation's blood supply. (2013, Jun 03). *Targeted News Service* Retrieved March 15, 2015, from <http://ezproxy.aut.ac.nz/login?url=http://search.proquest.com/docview/1362695661?accountid=8440>

- Rodriguez, L. M., Gallagher, D. R., Gallagher, M. W., Fisher, B. H., Humphries, J. R., & Malocha, D. C. (2014). Wireless SAW Sensor Temperature Extraction Precision. *Sensors Journal*, IEEE, 14(11), 3830-3837. doi: 10.1109/JSEN.2014.2344972
- Roussos, G. (2008). What is RFID Networked RFID (pp. 1-9): Springer London.
- Rundh, B. (2008). Radio frequency identification (RFID). *Marketing Intelligence & Planning*, 26(1), 97-114. doi: doi:10.1108/02634500810847174
- Saldanha, N., & Malocha, D. C. (2012). Pseudo-orthogonal frequency coded wireless SAW RFID temperature sensor tags. *Ultrasonics, Ferroelectrics, and Frequency Control*, IEEE Transactions on, 59(8), 1750-1758. doi: 10.1109/TUFFC.2012.2379
- Schuster, E. W., & Brock, D. L. (2007). Hardware: RFID Tags and Readers *Global RFID* (pp. 15-28): Springer Berlin Heidelberg.
- SignalR. (n.d.). Retrieved March 15, 2015, from <http://www.asp.net/signalr>
- SqlDependency Class. (n.d.). Retrieved March 13, 2015, from [https://msdn.microsoft.com/en-us/library/system.data.sqlclient.sqldependency\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.data.sqlclient.sqldependency(v=vs.110).aspx)
- SQL Server Service Broker. (n.d.). Retrieved March 10, 2015, from <https://msdn.microsoft.com/en-us/library/bb522893.aspx>
- Taralekar, R. (2013, May 14). Introduction to SignalR. Retrieved March 15, 2015, from <http://www.codeproject.com/Tips/590660/Introduction-to-SignalR>
- Understanding SQL Dependencies. (n.d.). Retrieved March 12, 2015, from [https://technet.microsoft.com/en-us/library/ms345449\(v=sql.105\).aspx](https://technet.microsoft.com/en-us/library/ms345449(v=sql.105).aspx)
- Vaishnavi, V. K., & William Kuechler, J. (2007). *Design Science Research Methods and Patterns: Innovating Information and Communication Technology*: Auerbach Publications.
- Valdez, G. A. (2012, December 17). SignalR: Building real time web applications. Retrieved March 15, 2015, from <http://blogs.msdn.com/b/webdev/archive/2012/12/17/signalr-building-real-time-web-applications.aspx>
- Venable, J., Pries-Heje, J., & Baskerville, R. (2012). A Comprehensive Framework for Evaluation in Design Science Research. In K. Peffers, M. Rothenberger & B. Kuechler (Eds.), *Design*

- Science Research in Information Systems. Advances in Theory and Practice (Vol. 7286, pp. 423-438): Springer Berlin Heidelberg.
- Virtanen, J., Yang, F., Ukkonen, L., Elsherbeni, A. Z., Babar, A. A., & Sydänheimo, L. (2014). Dual port temperature sensor tag for passive UHF RFID systems. *Sensor Review*, 34(2), 154-169. doi: doi:10.1108/SR-12-2011-681
- Visual C# resources. (n.d.). Retrieved March 17, 2015, from <https://msdn.microsoft.com/en-us/vstudio/hh341490.aspx>
- Visual Studio News. (n.d.). Retrieved March 16, 2015, from <https://msdn.microsoft.com/en-us/vstudio>
- vom Brocke, J., & Lippe, S. (2010). Taking a Project Management Perspective on Design Science Research. In R. Winter, J. L. Zhao & S. Aier (Eds.), *Global Perspectives on Design Science Research* (Vol. 6105, pp. 31-44): Springer Berlin Heidelberg.
- Wang, W., Xue, X., Huang, Y., & Liu, X. (2014). A Novel Wireless and Temperature-Compensated SAW Vibration Sensor. *Sensors* (Basel, Switzerland), 14(11), 20702-20712. doi: 10.3390/s141120702
- Want, R. (2006). An introduction to RFID technology. *Pervasive Computing, IEEE*, 5(1), 25-33. doi: 10.1109/MPRV.2006.2
- Wieringa, R. (2010). Design science methodology: principles and practice. Paper presented at the Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 2, Cape Town, South Africa.
- Winter, R. (2008). Design science research in europe. *European Journal of Information Systems*, 17(5), 470-475. doi:http://dx.doi.org/10.1057/ejis.2008.44
- Wray, B. R. (2011). Automation in blood banking: RFID. *Medical Laboratory Observer*, 43(10), 34. Retrieved March 15, 2015, from <http://ezproxy.aut.ac.nz/login?url=http://search.proquest.com/docview/898601712?accountid=8440>

- Yao, W., Chu, C.-H., & Li, Z. (2012). The Adoption and Implementation of RFID Technologies in Healthcare: A Literature Review. *Journal of Medical Systems*, 36(6), 3507-3525. doi: 10.1007/s10916-011-9789-8
- Zhang, X., Li, C., Gao, X., & Li, L. (2014). Integration Design of Temperature Sensor and Double RFID Tag. *Open Electrical & Electronic Engineering Journal*, 8.
- Zhu, Y., Jiang, W., Zhang, Q., & Guan, H. (2014). Energy-Efficient Identification in Large-Scale RFID Systems with Handheld Reader. *Parallel and Distributed Systems, IEEE Transactions on*, 25(5), 1211-1222. doi: 10.1109/TPDS.2013.175
- Zito, F., Aquilino, F., Fragomeni, L., Merenda, M., & Corte, F. G. D. (2010). CMOS wireless temperature sensor with integrated radiating element. *Sensors and Actuators A: Physical*, 158(2), 169-175. doi: <http://dx.doi.org/10.1016/j.sna.2009.12.014>

Appendix A – Sample Temperature Report

Title: Temperature Report

Description: This is a temperature report for bag id 5.

Blood Bag ID: 5

Event ID: 8	Time Stamp: 1:57:13 PM	Temperature: 4.00
Event ID: 9	Time Stamp: 1:57:18 PM	Temperature: 5.00
Event ID: 10	Time Stamp: 1:57:23 PM	Temperature: 0.00
Event ID: 11	Time Stamp: 1:57:28 PM	Temperature: 1.00
Event ID: 12	Time Stamp: 1:57:33 PM	Temperature: 2.00
Event ID: 13	Time Stamp: 1:57:38 PM	Temperature: 2.00
Event ID: 14	Time Stamp: 1:57:43 PM	Temperature: 7.00
Event ID: 15	Time Stamp: 1:57:48 PM	Temperature: 8.00
Event ID: 16	Time Stamp: 1:57:53 PM	Temperature: 3.00
Event ID: 17	Time Stamp: 1:57:58 PM	Temperature: 4.00
Event ID: 18	Time Stamp: 1:58:03 PM	Temperature: 5.00
Event ID: 19	Time Stamp: 1:58:08 PM	Temperature: 5.00
Event ID: 20	Time Stamp: 1:58:13 PM	Temperature: 0.00
Event ID: 21	Time Stamp: 1:58:18 PM	Temperature: 9.00
Event ID: 22	Time Stamp: 1:58:23 PM	Temperature: 1.00
Event ID: 23	Time Stamp: 1:58:28 PM	Temperature: 2.00

Event ID: 24	Time Stamp: 1:58:33 PM	Temperature: 3.00
Event ID: 25	Time Stamp: 1:58:38 PM	Temperature: 8.00
Event ID: 26	Time Stamp: 1:58:43 PM	Temperature: 9.00
Event ID: 27	Time Stamp: 1:58:48 PM	Temperature: 4.00
Event ID: 28	Time Stamp: 1:58:53 PM	Temperature: 6.00
Event ID: 29	Time Stamp: 1:58:58 PM	Temperature: 7.00
Event ID: 30	Time Stamp: 1:59:14 PM	Temperature: 9.00
Event ID: 31	Time Stamp: 2:01:35 PM	Temperature: 7.00
Event ID: 32	Time Stamp: 2:01:40 PM	Temperature: 2.00
Event ID: 33	Time Stamp: 2:01:45 PM	Temperature: 3.00
Event ID: 34	Time Stamp: 2:01:50 PM	Temperature: 3.00
Event ID: 35	Time Stamp: 2:01:55 PM	Temperature: 8.00
Event ID: 36	Time Stamp: 2:02:00 PM	Temperature: 9.00
Event ID: 37	Time Stamp: 2:02:05 PM	Temperature: 9.00
Event ID: 38	Time Stamp: 2:02:10 PM	Temperature: 0.00
Event ID: 39	Time Stamp: 2:02:15 PM	Temperature: 1.00
Event ID: 40	Time Stamp: 2:02:20 PM	Temperature: 6.00
Event ID: 41	Time Stamp: 2:02:25 PM	Temperature: 8.00
Event ID: 42	Time Stamp: 2:02:30 PM	Temperature: 9.00
Event ID: 43	Time Stamp: 2:02:35 PM	Temperature: 4.00
Event ID: 44	Time Stamp: 2:02:40 PM	Temperature: 5.00
Event ID: 45	Time Stamp: 2:02:45 PM	Temperature: 6.00
Event ID: 46	Time Stamp: 2:02:50 PM	Temperature: 7.00

Event ID: 47	Time Stamp: 2:02:55 PM	Temperature: 7.00
Event ID: 48	Time Stamp: 2:03:00 PM	Temperature: 8.00
Event ID: 49	Time Stamp: 2:03:05 PM	Temperature: 3.00
Event ID: 50	Time Stamp: 2:03:10 PM	Temperature: 4.00
Event ID: 51	Time Stamp: 2:03:15 PM	Temperature: 9.00
Event ID: 52	Time Stamp: 2:03:20 PM	Temperature: 9.00
Event ID: 53	Time Stamp: 2:03:25 PM	Temperature: 0.00
Event ID: 54	Time Stamp: 2:03:30 PM	Temperature: 5.00
Event ID: 55	Time Stamp: 2:03:35 PM	Temperature: 6.00
Event ID: 56	Time Stamp: 2:03:40 PM	Temperature: 8.00
Event ID: 57	Time Stamp: 2:03:45 PM	Temperature: 7.00
Event ID: 58	Time Stamp: 2:03:50 PM	Temperature: 8.00
Event ID: 59	Time Stamp: 2:03:55 PM	Temperature: 0.00
Event ID: 60	Time Stamp: 2:04:00 PM	Temperature: 1.00
Event ID: 61	Time Stamp: 2:04:05 PM	Temperature: 2.00
Event ID: 62	Time Stamp: 2:04:10 PM	Temperature: 7.00
Event ID: 63	Time Stamp: 2:04:15 PM	Temperature: 8.00
Event ID: 64	Time Stamp: 2:04:20 PM	Temperature: 3.00
Event ID: 65	Time Stamp: 2:04:25 PM	Temperature: 5.00
Event ID: 66	Time Stamp: 2:04:30 PM	Temperature: 6.00
Event ID: 67	Time Stamp: 2:04:35 PM	Temperature: 7.00
Event ID: 68	Time Stamp: 2:04:40 PM	Temperature: 7.00
Event ID: 69	Time Stamp: 2:04:45 PM	Temperature: 8.00

Event ID: 70	Time Stamp: 2:04:50 PM	Temperature: 9.00
Event ID: 71	Time Stamp: 2:04:55 PM	Temperature: 0.00
Event ID: 72	Time Stamp: 2:05:00 PM	Temperature: 5.00
Event ID: 73	Time Stamp: 2:05:05 PM	Temperature: 7.00
Event ID: 74	Time Stamp: 2:05:10 PM	Temperature: 8.00
Event ID: 75	Time Stamp: 2:05:15 PM	Temperature: 9.00
Event ID: 76	Time Stamp: 2:05:20 PM	Temperature: 4.00
Event ID: 77	Time Stamp: 2:05:25 PM	Temperature: 5.00
Event ID: 78	Time Stamp: 2:05:30 PM	Temperature: 7.00
Event ID: 79	Time Stamp: 2:05:35 PM	Temperature: 8.00
Event ID: 80	Time Stamp: 2:05:40 PM	Temperature: 7.00
Event ID: 81	Time Stamp: 2:05:45 PM	Temperature: 9.00
Event ID: 82	Time Stamp: 2:05:50 PM	Temperature: 0.00
Event ID: 83	Time Stamp: 2:05:55 PM	Temperature: 1.00
Event ID: 84	Time Stamp: 2:06:00 PM	Temperature: 2.00
Event ID: 85	Time Stamp: 2:06:05 PM	Temperature: 4.00
Event ID: 86	Time Stamp: 2:06:10 PM	Temperature: 9.00
Event ID: 87	Time Stamp: 2:06:15 PM	Temperature: 0.00
Event ID: 88	Time Stamp: 2:06:20 PM	Temperature: 9.00
Event ID: 89	Time Stamp: 2:06:25 PM	Temperature: 1.00
Event ID: 90	Time Stamp: 2:06:30 PM	Temperature: 6.00
Event ID: 91	Time Stamp: 2:06:35 PM	Temperature: 7.00
Event ID: 92	Time Stamp: 2:06:40 PM	Temperature: 8.00

Event ID: 93	Time Stamp: 2:06:45 PM	Temperature: 9.00
Event ID: 94	Time Stamp: 2:06:50 PM	Temperature: 1.00
Event ID: 95	Time Stamp: 2:06:55 PM	Temperature: 2.00
Event ID: 96	Time Stamp: 2:07:00 PM	Temperature: 7.00
Event ID: 97	Time Stamp: 2:07:05 PM	Temperature: 8.00
Event ID: 98	Time Stamp: 2:07:10 PM	Temperature: 9.00
Event ID: 99	Time Stamp: 2:07:15 PM	Temperature: 9.00
Event ID: 100	Time Stamp: 2:07:20 PM	Temperature: 0.00
Event ID: 101	Time Stamp: 2:07:25 PM	Temperature: 1.00
Event ID: 102	Time Stamp: 2:07:30 PM	Temperature: 6.00
Event ID: 103	Time Stamp: 2:07:35 PM	Temperature: 8.00
Event ID: 104	Time Stamp: 2:07:40 PM	Temperature: 9.00
Event ID: 105	Time Stamp: 2:07:45 PM	Temperature: 0.00
Event ID: 106	Time Stamp: 2:07:50 PM	Temperature: 1.00
Event ID: 107	Time Stamp: 2:07:55 PM	Temperature: 3.00
Event ID: 108	Time Stamp: 2:08:00 PM	Temperature: 8.00
Event ID: 109	Time Stamp: 2:08:05 PM	Temperature: 9.00
Event ID: 110	Time Stamp: 2:08:10 PM	Temperature: 0.00
Event ID: 111	Time Stamp: 2:08:15 PM	Temperature: 1.00
Event ID: 112	Time Stamp: 2:08:20 PM	Temperature: 3.00
Event ID: 113	Time Stamp: 2:08:25 PM	Temperature: 4.00
Event ID: 114	Time Stamp: 2:08:30 PM	Temperature: 3.00
Event ID: 115	Time Stamp: 2:08:35 PM	Temperature: 5.00

Event ID: 116	Time Stamp: 2:08:40 PM	Temperature: 6.00
Event ID: 117	Time Stamp: 2:08:45 PM	Temperature: 7.00
Event ID: 118	Time Stamp: 2:08:50 PM	Temperature: 2.00
Event ID: 119	Time Stamp: 2:08:55 PM	Temperature: 3.00
Event ID: 120	Time Stamp: 2:09:00 PM	Temperature: 5.00
Event ID: 121	Time Stamp: 2:09:05 PM	Temperature: 4.00
Event ID: 122	Time Stamp: 2:09:10 PM	Temperature: 9.00
Event ID: 123	Time Stamp: 2:09:15 PM	Temperature: 0.00
Event ID: 124	Time Stamp: 2:09:20 PM	Temperature: 5.00
Event ID: 125	Time Stamp: 2:09:25 PM	Temperature: 7.00
Event ID: 126	Time Stamp: 2:09:30 PM	Temperature: 8.00
Event ID: 127	Time Stamp: 2:09:35 PM	Temperature: 9.00
Event ID: 128	Time Stamp: 2:09:40 PM	Temperature: 0.00
Event ID: 129	Time Stamp: 2:09:45 PM	Temperature: 2.00
Event ID: 130	Time Stamp: 2:09:50 PM	Temperature: 1.00
Event ID: 131	Time Stamp: 2:09:55 PM	Temperature: 2.00
Event ID: 132	Time Stamp: 2:10:00 PM	Temperature: 7.00
Event ID: 133	Time Stamp: 2:10:05 PM	Temperature: 9.00
Event ID: 134	Time Stamp: 2:10:10 PM	Temperature: 3.00
Event ID: 135	Time Stamp: 2:10:15 PM	Temperature: 5.00
Event ID: 136	Time Stamp: 2:10:21 PM	Temperature: 4.00
Event ID: 137	Time Stamp: 2:10:26 PM	Temperature: 6.00
Event ID: 138	Time Stamp: 2:10:31 PM	Temperature: 0.00

Event ID: 139	Time Stamp: 2:10:36 PM	Temperature: 2.00
Event ID: 140	Time Stamp: 2:10:41 PM	Temperature: 1.00
Event ID: 141	Time Stamp: 2:10:46 PM	Temperature: 3.00
Event ID: 142	Time Stamp: 2:10:51 PM	Temperature: 4.00
Event ID: 143	Time Stamp: 2:10:56 PM	Temperature: 5.00
Event ID: 144	Time Stamp: 2:11:01 PM	Temperature: 6.00
Event ID: 145	Time Stamp: 2:11:06 PM	Temperature: 1.00
Event ID: 146	Time Stamp: 2:11:11 PM	Temperature: 3.00
Event ID: 147	Time Stamp: 2:11:16 PM	Temperature: 7.00
Event ID: 148	Time Stamp: 2:11:21 PM	Temperature: 9.00
Event ID: 149	Time Stamp: 2:11:26 PM	Temperature: 0.00
Event ID: 150	Time Stamp: 2:11:31 PM	Temperature: 1.00

Appendix B – Sample Temperature Worksheet

EventID	TimeStamp	Temperature	BloodBagID
8	15/02/2015 13:57	4	5
9	15/02/2015 13:57	5	5
10	15/02/2015 13:57	0	5
11	15/02/2015 13:57	1	5
12	15/02/2015 13:57	2	5
13	15/02/2015 13:57	2	5
14	15/02/2015 13:57	7	5
15	15/02/2015 13:57	8	5
16	15/02/2015 13:57	3	5
17	15/02/2015 13:57	4	5
18	15/02/2015 13:58	5	5
19	15/02/2015 13:58	5	5
20	15/02/2015 13:58	0	5
21	15/02/2015 13:58	9	5
22	15/02/2015 13:58	1	5
23	15/02/2015 13:58	2	5
24	15/02/2015 13:58	3	5
25	15/02/2015 13:58	8	5
26	15/02/2015 13:58	9	5
27	15/02/2015 13:58	4	5
28	15/02/2015 13:58	6	5

29	15/02/2015 13:58	7	5
30	15/02/2015 13:59	9	5
31	15/02/2015 14:01	7	5
32	15/02/2015 14:01	2	5
33	15/02/2015 14:01	3	5
34	15/02/2015 14:01	3	5
35	15/02/2015 14:01	8	5
36	15/02/2015 14:02	9	5
37	15/02/2015 14:02	9	5
38	15/02/2015 14:02	0	5
39	15/02/2015 14:02	1	5
40	15/02/2015 14:02	6	5
41	15/02/2015 14:02	8	5
42	15/02/2015 14:02	9	5
43	15/02/2015 14:02	4	5
44	15/02/2015 14:02	5	5
45	15/02/2015 14:02	6	5
46	15/02/2015 14:02	7	5
47	15/02/2015 14:02	7	5
48	15/02/2015 14:03	8	5
49	15/02/2015 14:03	3	5
50	15/02/2015 14:03	4	5
51	15/02/2015 14:03	9	5

52	15/02/2015 14:03	9	5
53	15/02/2015 14:03	0	5
54	15/02/2015 14:03	5	5
55	15/02/2015 14:03	6	5
56	15/02/2015 14:03	8	5
57	15/02/2015 14:03	7	5
58	15/02/2015 14:03	8	5
59	15/02/2015 14:03	0	5
60	15/02/2015 14:04	1	5
61	15/02/2015 14:04	2	5
62	15/02/2015 14:04	7	5
63	15/02/2015 14:04	8	5
64	15/02/2015 14:04	3	5
65	15/02/2015 14:04	5	5
66	15/02/2015 14:04	6	5
67	15/02/2015 14:04	7	5
68	15/02/2015 14:04	7	5
69	15/02/2015 14:04	8	5
70	15/02/2015 14:04	9	5
71	15/02/2015 14:04	0	5
72	15/02/2015 14:05	5	5
73	15/02/2015 14:05	7	5
74	15/02/2015 14:05	8	5

75	15/02/2015 14:05	9	5
76	15/02/2015 14:05	4	5
77	15/02/2015 14:05	5	5
78	15/02/2015 14:05	7	5
79	15/02/2015 14:05	8	5
80	15/02/2015 14:05	7	5
81	15/02/2015 14:05	9	5
82	15/02/2015 14:05	0	5
83	15/02/2015 14:05	1	5
84	15/02/2015 14:06	2	5
85	15/02/2015 14:06	4	5
86	15/02/2015 14:06	9	5
87	15/02/2015 14:06	0	5
88	15/02/2015 14:06	9	5
89	15/02/2015 14:06	1	5
90	15/02/2015 14:06	6	5
91	15/02/2015 14:06	7	5
92	15/02/2015 14:06	8	5
93	15/02/2015 14:06	9	5
94	15/02/2015 14:06	1	5
95	15/02/2015 14:06	2	5
96	15/02/2015 14:07	7	5
97	15/02/2015 14:07	8	5

98	15/02/2015 14:07	9	5
99	15/02/2015 14:07	9	5
100	15/02/2015 14:07	0	5
101	15/02/2015 14:07	1	5
102	15/02/2015 14:07	6	5
103	15/02/2015 14:07	8	5
104	15/02/2015 14:07	9	5
105	15/02/2015 14:07	0	5
106	15/02/2015 14:07	1	5
107	15/02/2015 14:07	3	5
108	15/02/2015 14:08	8	5
109	15/02/2015 14:08	9	5
110	15/02/2015 14:08	0	5
111	15/02/2015 14:08	1	5
112	15/02/2015 14:08	3	5
113	15/02/2015 14:08	4	5
114	15/02/2015 14:08	3	5
115	15/02/2015 14:08	5	5
116	15/02/2015 14:08	6	5
117	15/02/2015 14:08	7	5
118	15/02/2015 14:08	2	5
119	15/02/2015 14:08	3	5
120	15/02/2015 14:09	5	5

121	15/02/2015 14:09	4	5
122	15/02/2015 14:09	9	5
123	15/02/2015 14:09	0	5
124	15/02/2015 14:09	5	5
125	15/02/2015 14:09	7	5
126	15/02/2015 14:09	8	5
127	15/02/2015 14:09	9	5
128	15/02/2015 14:09	0	5
129	15/02/2015 14:09	2	5
130	15/02/2015 14:09	1	5
131	15/02/2015 14:09	2	5
132	15/02/2015 14:10	7	5
133	15/02/2015 14:10	9	5
134	15/02/2015 14:10	3	5
135	15/02/2015 14:10	5	5
136	15/02/2015 14:10	4	5
137	15/02/2015 14:10	6	5
138	15/02/2015 14:10	0	5
139	15/02/2015 14:10	2	5
140	15/02/2015 14:10	1	5
141	15/02/2015 14:10	3	5
142	15/02/2015 14:10	4	5
143	15/02/2015 14:10	5	5

144	15/02/2015 14:11	6	5
145	15/02/2015 14:11	1	5
146	15/02/2015 14:11	3	5
147	15/02/2015 14:11	7	5
148	15/02/2015 14:11	9	5
149	15/02/2015 14:11	0	5
150	15/02/2015 14:11	1	5