ETLA ELINKEINOELÄMÄN TUTKIMUSLAITOS
THE RESEARCH INSTITUTE OF THE FINNISH ECONOMY

Keskusteluaiheita
Discussion Papers
24 May 2011

No **1250**

Wireless Acquisition of

Process Data

Ye Zhang*

* ETLA – Elinkeinoelämän Tutkimuslaitos, ye.zhang@etla.fi

Acknowledgements:
This thesis topic is supported by Value Creation in Smart Living Environments for Senior Citizens (VESC) project, which is funded by the Academy of Finland. This report is part of the 'ICT, Service Innovations and Productivity' project in ETLA. Funding from the Technology Industries of Finland Centennial Foundation is kindly acknowledged.

ISSN 0781–6847

Contents

Abstract

This study discusses a novel method called automatic process measurement, which is based on the idea of mining process data from workflow logs. We improve the process mining technique by using Bluetooth wireless technology to do real-time acquisition of process data. The automatic measurement system is capable of collecting process data of elderly people's daily process as well as nursing personnel's behavior in the open healthcare. Similarly, retail and logistics processes can be measured with the system. The data can be used further in process analysis and modeling.

Key words: Real-time data acquisition, automatic process measurement, process mining, Bluetooth

JEL: D22, C81, I112

Tiivistelmä

Tässä tutkimuksessa tarkastellaan automaattista työprosessien mittaamista käyttäen hyväksi tiedonlouhintaa. Mittaustieto kerätään reaaliajassa Bluetooth-antureista ja saatu lokitieto analysoidaan. Automaattisella mittausjärjestelmällä voidaan seurata esimerkiksi vanhusten päivittäistä dataa sekä terveydenhuoltohenkilöstön prosessidataa avohoidossa. Samoin kaupan ja logistiikan prosesseja voidaan mitata järjestelmää käyttäen. Mitattua dataa voidaan käyttää prosessien analysointiin ja mallinnukseen.

Asiasanat: Reaaliaikainen datan keräys, automaattinen prosessien mittaus, tiedon louhinta prosesseista, Bluetooth

1 Introduction

This thesis topic is a part of Value Creation in Smart Living Environment for Senior Citizen (VESC) project, which is founded by the Academy of Finland, and is planned to extend to the end of 2012. The project is conducted in cooperation with the Department of Information Processing Science (TOL), Department of Medical Technology (LTEK) and Department of Architecture (ARK) of the University of Oulu, VTT Technical Research Center, and the Oulu Deaconess Institute. The main goal of the VESC project is to support the elderly in their daily activities and improve the quality of the elderly healthcare by developing technology-based smart services (Academy of Finland, 2011). This chapter delineates the study background, research purpose, and the research methods.

1.1 Research background

As human society entered the 21st century, the world's population aging problem has arisen: medicine has made great progress in modem society, which extends people's life expectancy, leads to the augmenting number of elderly. By the statistical standard of the United Nations, if people over 60 exceed 10 percent of the population, or people over 65 exceed seven percent of the population, then it is so-called 'Old Age Society'. The population aging is a global and unalterable trend, according to the world population aging prospects conducted by (Population Division of the United Nations, 2010): the proportion of old people has come up to 11 percent by 2009, three percent higher than in 1950; developing countries have a faster growth rate than developed countries, and it predicts that the proportion of old people will reach 22 percent in 2050.

The aging of population has put severer challenges to the whole world, requires the concern and efforts of the whole society. But, we're facing the problems of lacking productive workforce such as nursing personnel, to support the social healthcare and other services (Hori, Nishida, Aizawa, Murakami, & Mizoguchi, 2004). In order to create a more comfort, better and securer life for the elderly under this circumstance, a lot of studies on automatic elderly assistant systems have been conducted: (Chan, Hariton, Ringeard, & Campo, 1995) conducted studies on learning the habits of old person, and diagnose their behavior changes by using multi-sensor system and artificial neural networks; mobile robotic assistant systems, which remind old people about routine activities in their daily life, such as take medicine and eat, and give them guidance in their surround environments as well (Pollack et al., 2002); real-time monitoring systems, (Noury, 2002) developed a real-time detection of the fall of the elderly by using tilt sensors and wireless communication technique; (Hwang, Kang, Jang, & Kim, 2004) proposed another similar real-time fall detection system with Bluetooth module and use accelerometer, gyroscope and tilt sensor simultaneously, (Hori, Nishida, Aizawa, Murakami, & Mozoguchi, 2004) developed an ultrasonic sensor network system that is used in nursing room to monitor elderly people; smart healthcare or home automation systems, for instance the B-Live home automation system that is implemented in 2007 by Santos, Bartolomeu, Fonseca, & Mota.

To a certain extent, those systems are much less intrusive, provide continuous real-time monitoring of the health condition of the elderly people as well as emergency accidents. However, as more precise follow-up and better service quality is eminently demanded, those studies are not sufficient. None of them consider to carry out the measurement of nursing personnel's be-

havior in the open healthcare process, nor do they combine developing smart living environment with the concept of process management.

This thesis combines together the concepts of smart living environment and process management. Since the concept of process management was advanced in the 1990s, it has become practitioners and academicians' focus of concern. Usually, it is applied in business and manufacturing fields to control and optimize an organization's business processes. A defined business process will create more efficiency and profit. Thus, this study uses process management concept to ensure better quality of the elderly healthcare.

A precise model of the process is of paramount importance for more effective management of the elderly healthcare process. A precise model can efficiently presents the abstraction of the real world, thus enables better understanding and systematic analysis of the process. This thesis adopts the idea of process mining technique, which is widely used for process management in recent years, it extracts information from event logs and then generate process models automatically. However, those event logs are usually recorded by various information systems, the accuracy of the data cannot be ensured. So this thesis proposes a novel method called automatic process measurement, which uses wireless technologies to collect the data of the elderly healthcare process for process mining and process analysis.

A great many studies on real-time monitoring by using wireless sensor networks have been done in many areas: (Mainwaring, Polastre, Szewezyk, Culler, & Anderson, 2002) implemented a real-world habitat monitoring system based on tiered sensor network architecture and WLAN communication technology, the network with 32 nodes was deployed on a small island off the coast of Maine, aim to run for nine months from non-rechargeable power sources to monitoring seabird nesting environment and behavior; (Jovanov et al., 2003) also used WLAN to develop a distributed wireless system to communicate with personal health monitors, those individual monitors are based on intelligent sensors that enable noninvasive, unobtrusive medical monitoring during normal activities; (Xu, Rangwala, Chintalapudi, et al., 2004) applied wireless sensor network in structural health monitoring, they designed a wireless sensor network system called Wisden for structural data acquisition, the system seeks to detect localize damage in buildings, bridges; (Park, Liu, & Chou, 2005) introduced an ultra-compact low-power wireless sensor node for real-time motion monitoring, called Eco, it is suitable for many applications (such as medicine monitoring, intelligent environment, and new computer-human interface); (He, Lu, & Abdelzaher, 2003) conducted one research on real-time communication protocol for sensor networks, they studied the SPEED protocol, which is tailored to be efficient for resource-scarce sensor networks. So this thesis also uses wireless technologies in the elderly healthcare field, instead of WLAN, we adopt Bluetooth technology that is capable of collecting real-time activity data for automatic process measurement.

1.2 Research purpose & research problems

The objective of VESC project is to support the elderly in their daily activities and improve the quality of the elderly healthcare by developing technology-based smart services. The technology used should play a role that motivates, activates and empowers the elderly to live more independent lives, instead of a role of controller (Academy of Finland, 2011), so in VESC research project, there are needs to measure:

– Patient's indoor living process.
– How to do acquisition of real-time data for automatic process measurement?

This research has been conducted to develop and test a prototype for these needs of measurement. Since the technologies used should be less intrusive, we contemplate the automatic process measurement method. Usually, the automatic process measurement approach uses event logs that are recorded by information systems, and each log should at least contains: activity ID, activity name, timestamps (begin time and end time), user ID. In the case of modeling a healthcare process, various activities and data are distributed at different locations, so this research deliberates data acquisition in wireless sensor network, and focus on solving the following two research questions:

– How to do acquisition of real-time data for automatic process measurement?
– Which kinds of Indoor Communication Technologies are most suitable?

For automatic process measurement, it needs the following real-time data: who performed the activity, including the type of participant (nurse or patient), and the identity to distinguish different nurses or patients; duration of the activity, like when does the activity begin and when it is ended, which are used to analyze the sequence of activities; what is the activity. Besides, to achieve the combination of real-time healthcare monitoring and the support for process management, it also requires to record the information of a patient's health status.

This thesis adopts wireless indoor communication technologies, there are wide selections, like 802.11 WLAN, Bluetooth, ZigBee. Wi-Fi technology is the wireless local area network (WLAN) that based on IEEE 802.11 standards, it enables the connection to the Internet through access points. Current usages of Wi-Fi including Internet access, Citywide/Campus-wide Wi-Fi, and Direct computer-to-computer communications. Bluetooth is a short-range wireless communication technology standard, which is maintained by the Bluetooth Special Interest Group. It implements the network interconnection between fixed devices or mobile devices within a short distance, it has the advantages of low power consumption, lower imple-

Table 1 Wireless technologies comparison

Type	*Wi-Fi*	*Bluetooth*	*Zigbee*
Range	50-100 m	10 m	10-100 m
Operating Frequency	2.4 and 5 GHz	2.4 GHz	868 MHz (Europe); 900-928 MHz (NA); 2.4 GHz (worldwide)
Complexity (Device and application impact)	Very high	High	Low
Power Consumption (Battery option and life)	High	Medium	Very low
Security	64 and 128 bit encryption		128 AES plus application layer security
Data rate	11 & 54 Mbits/sec	1 Mbits/s	20, 40, and 250 Kbits/s
Implementation Cost	High	Low	Low

(Jin-Shyan Lee, Yu-Wei Su, & Chung-Chou Shen, 2007).

mentation cost, and security. Zigbee is a low-bandwidth wireless communication technology based on the IEEE 802.15.4. It has been widely used in: lighting control, environment control, central air conditioning system, building automation system. Table 1 shows the comparison of these wireless technologies.

In this study, we focus on using Bluetooth communication technology to develop the measurement system, and the system is targeted to Linux-based mobile devices. Another objective of this study is to evaluate whether the system fulfill the measurement needs through laboratory and practical environment testing. In order to better support prototype testing, friendly and simple graphical user interface is implemented to represent real-time workflow.

1.3 Research method

To conduct this study, we use three research methods: literature research, constructive research, and case study. According to the research method used, the whole research process is divided into three phases.

Phase one, the study starts with literature research. Searching for new literature about solutions for real-time data acquisition, as well as wireless communication technologies. Thus, understand the research problems comprehensively and accurately, and amass materials that hold different views. Base on these, further investigation, comparison and analysis can be conducted.

Phase two, after gain sufficient understanding of the research problems, we continue the study with the most common research method for computer science – constructive research method. First, it should conduct more related research before develop the measurement system, it focuses on the selected wireless communication technology and its protocols, the selection of programming languages, operating systems (OS), development tools, and graphical user interface design. Second, analyze requirements of solving the research problems, then begin develop the measurement system.

Phase three, valuation should be addressed after the prototype is developed. First, perform the laboratory testing in the Department of Information Processing Science (TOL), the University of Oulu, then conduct the practical environment testing in Helsinki University Central Hospital (HUS).

1.4 Structure of this document

The following sections elaborate the theoretical background of the research problems, and introduce the technologies used as well as the solution architecture. The rest of this thesis organized as follows:

Chapter 2 introduces the concepts of process, process modeling, process mining, outlines the needs of measure the activities of elderly people in their daily lives, and the activities of nurses in the open healthcare processes. In addition, it illustrates the automatic process measurement solution, which was adopted in this study.

Chapter 3 presents Bluetooth communication technology, discusses the advantages and usage of Bluetooth, and introduces the history of Bluetooth technology. It gives a detailed view of general Bluetooth protocol stack as well as specific Bluetooth protocol stack for Linux.

Chapter 4 discusses the selection of targeted platforms and development tools. It first presents three targeted platforms for running the measurement system – Ubuntu Netbook Remix, Maemo 5, Android 2.1. Then, it introduces development tools, such as Qt Creator, Nokia Qt Software Development Kit (SDK), Eclipse, Android Development Tools (ADT), which are used to develop the system for different platforms.

Chapter 5 depicts the design of the measurement system in detail. It discusses the architecture of the measurement system, the development process. It uses Use Case diagram to determine the requirements of the system, and it uses Class diagram, Activity diagram, Sequence diagram to present the structure and application logic of the measurement system. It also delineates the design of the system's graphical user interface, and compares developed prototypes, targeted platforms, development tools, and different Bluetooth APIs in those platforms.

Chapter 6 discusses the laboratory and practical environment testing, including: testing environment, testing method, testing procedure, testing results, problems of the developed prototypes and solutions.

Chapter 7 presents the conclusions of this research and further research plans.

2 Process measurement

There are needs to study the behaviors of the elderly, and to measure the behaviors of nurses. This Chapter introduces the concepts of process, process modeling, and process mining. It discusses the automatic process measurement method, which is based on the idea of mining processes from event logs.

2.1 Definition of process

There are various definitions of process. In 1776, Adam Smith first divided the work into a set of simple tasks to improve productivity, and described the process as a collection of related, structured tasks. Later (Davenport, 1993) defined that a process is a specific ordering of work activities across time and space, and these activities should be structured, with a beginning and an ending, clear inputs and outputs. (Hammer & Champy, 1993) and (Rummler & Brache, 1995) gave the similar definitions as well, they defined the process as a collection of activities or a series of steps that take inputs and create value for customers.

Generally, a process comprises a series of interrelated tasks in order of precedence and together convert inputs of a process into outputs. It often refers to business process that begins with an objective (e.g. provides specific services or products to customers, and aim to achieve it at the end of the process). These decomposed activities can be computer-based, for instance using Information Systems (IS), or they can be performed manually. In a word, process emphasizes the importance of activities work as a whole to enhance the added value of a process.

Processes are classified into three categories. Higher-level management processes, which include corporate governance and strategic management in the operation of a company's business as a system. Operational processes, which comprise the most essential core business, (Rummler & Brache, 1995) delineated that this type of processes will result in services received by customers, so they also call these primary processes. Lower-level supporting processes, which give support to the core processes, (Rummler & Brache, 1995) thought these processes are essential, they provide invisible services to customers. This thesis studies on operational process in the elderly healthcare system, including elderly people's daily processes and open care processes.

2.2 Measurement needs

In order to develop technology-based smart services that are less intrusive and can support the elderly live more independent lives, there is a need to study the elderly people's daily behaviors. To improve the quality of the elderly healthcare, there is also a need to measure the behaviors of nurses in the open healthcare process.

For more effective management of the elderly healthcare process, a precise model of the process is of paramount importance. A precise model can efficiently present the abstraction of the real world, thus enables better understanding and systematic analysis of the process. There are various traditional process modeling approaches, however, they are time-consuming and require modelers with professional cognizance and the skill in process modeling. Models are defined by modelers manually based on the analysis of process documents and consultations with process participators. This leads to several problems, such as the individual experience of modelers and participators will greatly affect the objectivity of a process model, and large numbers of redundant data will influence the efficiency of traditional modeling methods (Yan Li & Yu-qiang Feng, 2006); in many domains, processes are evolving and people typically have an oversimplified incorrect view of the actual processes (Van Der Aalst, et al., 2011); usually, these approaches have difficulty of balancing usability and flexibility (Chang-rui Ren, et al., 2008).

Very detail analysis of various data is crucial. However, the healthcare process including the elderly as participants, most of them cannot even do simple daily activity, needless to say information communication and insure accurate data for process modeling. Therefore, the automatic process measurement method is highly demanded in order to study the behaviors of the elderly and open healthcare processes.

2.3 Automatic process measurement

Traditional process modeling methods usually use graphical modeling techniques, like flowchart that emerged in the 1920s, functional flow block diagram, control flow diagram (CFD), and the project evaluation and review technique (PERT) in the 1950s. In recent years, a process modeling method called process mining has been widely. In this study, we introduce a novel concept – automatic process measurement, which follows the same thought of process mining.

Process mining technique extracts information from event logs, and then generate process models automatically. More accurate data in the process logs, better the models will reflect the actual process. This idea of mining process information from workflow logs isn't new, it is applied firstly in software engineering area by Cook and Alexander as early as in 1995. And later in 1998, the idea was used in process modeling by Agrawal, R., Gunopulos, D., and Leymann, F., however, it was accepted as a new process modeling method only in recent years.

Based on the problems of traditional process modeling methods mentioned in section 2.2, process mining technique reduces the difficulty of process modeling and measurement. It realizes the automation of process modeling, extremely simplifies the procedure of analyzing and diagramming processes, insure the efficiency of process modeling. In addition, process mining technique improves the objectivity of process modeling. It is a dynamic mining method, it concerns on various tasks or activities, their states and transfer relationships, so accurate workflow logs will enable the process model reflects the actual process exactly.

The basis of process mining is the process logs, we define the following requirements of process logs:

- Recording detailed process data, at least includes the name of the activity, the name of the performer, duration of the activity, when does the activity begin and end, and other data related to the activity.
- Log data are correct, integrated, and accurate, can reflect the true processes.
- Log data should be able to be converted into constructed data easily, and assure data integrity during transformation.
- One activity can only have one beginning and one ending, which are recorded in log files in chronological sequence.

In this study, we focus on creating process logs with high accuracy, and solving our research questions: how to do real-time data collection for the needs of automatic process measurement. These logs record the activities of the elderly people in their daily lives as well as nurses' activities in open healthcare processes. Based on these activity data, we can study the behaviors of the elderly, thus, better smart services can be developed to support the elderly live more independent lives. Moreover, we can improve the quality of elderly health care through analyzing open care processes. Subsequent work could be automatic process modeling, which extracts process related information from event logs, and build process models. But, this is out of the scope of our research in this paper.

3 Bluetooth communication technology

Bluetooth communication technology has several advantages. It is global, it has high anti-interference capability, low power consumption and low implementation cost. So it is chosen for the measurement system. This chapter introduces Bluetooth technology, including its history and the Bluetooth protocol stack. In this study, the targeted platform is Linux-based platforms, thus, we also discuss the specific Bluetooth protocol stack for Linux platform.

3.1 Overview of Bluetooth

Bluetooth is a short-range wireless communication technology standard. Its initial aim is to eliminate most of the interconnect cables between various electronic devices, such as personal digital assistant (PDA), mobile phone, and laptop (Miller, 2001; Bisdikian, 2001, Kansal, 2002, Ferro & Potorti, 2005). It also simplifies the communication between these devices and Internet. Bluetooth is an open technology standard, using a worldwide license free radio band – Industrial, Scientific, and Medical band (ISM) at 2.4GHz, it has following characteristics:

Bluetooth is global. 2.4GHz band can be used worldwide, ISM band in most countries throughout the world ranges from 2.4GHz to 2.4835GHz. To use this band, it doesn't require the license from radio resource administration in each country.

High anti-interference capability. Among various wireless communication technologies, 2.4GHz is very popular. Cordless telephone, microwave oven, Bluetooth, Wireless Local Area Network (WLAN), wireless USB, and Zigbee all operate at 2.4GHz ISM band. Bluetooth technology uses Adaptive Frequency-hopping spread spectrum (AFH) as its solution to avoid interference from other technologies.

Small size, easy to integrate and low power consumption. In terms of recent usages of Bluetooth, it is not limited in computer peripheral equipment, it can be integrated into almost any digital equipment. Minimal power consumption is especially important for small size devices, Bluetooth has four operational modes after a connection is built: active mode, sniff mode, hold mode, park mode, the latter three are low power consumption modes.

Support transmitting voice and data simultaneously. It supports one asynchronous data channel, three synchronous voice channels, and one channel for both asynchronous data and voice simultaneous transmission (Sairam et al., 2002).

It enables temporary ad hoc connection. Piconet is established dynamically and automatically when one Bluetooth device enters the radio proximity of another Bluetooth device (Kansal, 2002). Maximum of eight Bluetooth devices can connect into a Piconet network simultaneously. The role of Bluetooth devices inside a Piconet can be a master or a slave, master is the one initiates the connecting request, each Piconet has only one master, but one Bluetooth device can belong to several other Piconet networks simultaneously by using Optical Code Division Multiplexing (OCDM) technology.

The Bluetooth Special Interest Group (SIG) is responsible for overseeing Bluetooth standard development, Bluetooth technology and trademark licensing (Wikipedia 2011), To popularize Bluetooth technology, SIG opens all Bluetooth technology standards, it enables anyone to do development on Bluetooth. In this study, we adopt Bluetooth communication technology because the implementation cost is low, and it has a high degree of portability, moreover, it is less harmful to human body.

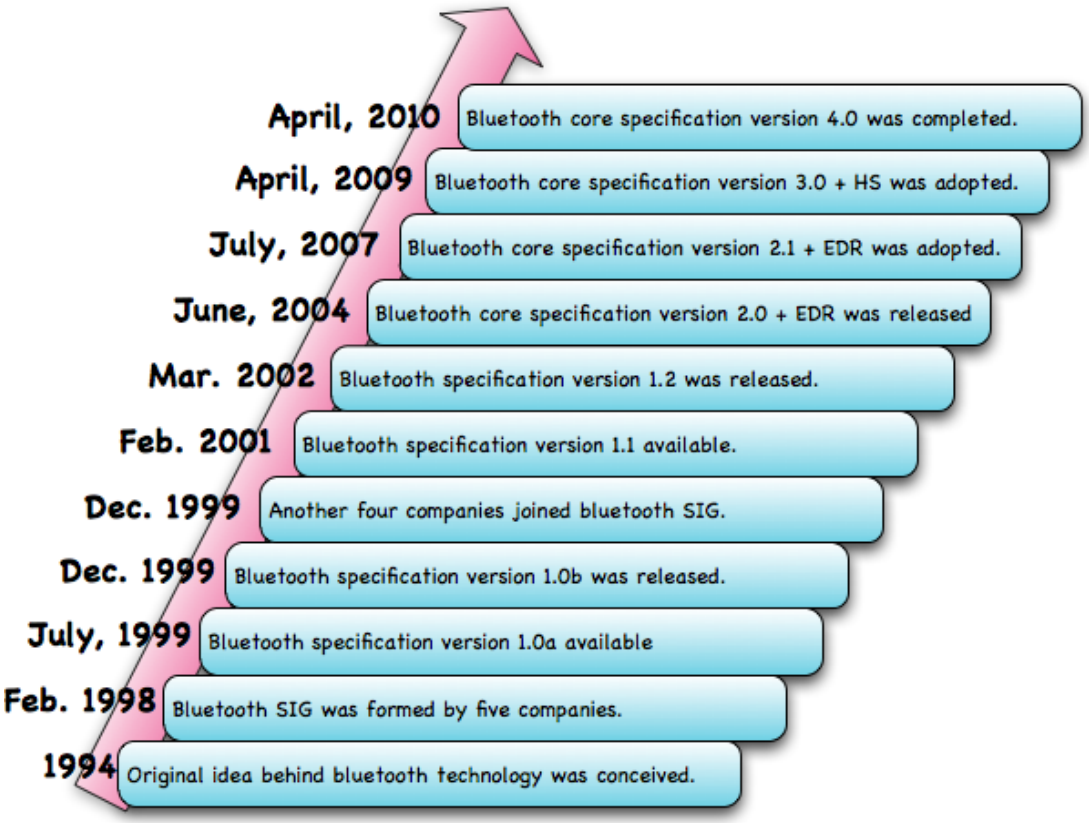
3.2 Bluetooth history

The name of Bluetooth comes from Harald Gormsoon, the king of Denmark in the 10th century. His teeth were often blue because he liked blueberry very much, so he was nicknamed 'Bluetooth'. Blueberry was considered inedible in those days because of its strange color. Thus, king's Bluetooth became the emblem of innovation and trying. Figure 1 shows the milestones of the development of Bluetooth technology.

The original idea behind Bluetooth technology was first developed by Ericsson Mobile Communications in 1994, they were searching low-power-consumption systems to substitute interconnect cables, which are used in short-range communication between their mobile phones and other relevant accessories (Ferro & Potorti, 2005). Ericsson hoped wireless communication technologies can unify their standards, so they named the technology 'Bluetooth'. And in 1998, Ericsson with other four companies (including IBM, Intel, Nokia, Toshiba) formed the Bluetooth SIG. In July 1999, Bluetooth specification version 1.0a was available for public, and later in December in the same year, Bluetooth specification version 1.0b was released and the SIG group added another four member companies (including 3COM, Agere, Microsoft and Motorola).

In the following years, six versions of Bluetooth specifications have been developed: in February 2001 and March 2002, Bluetooth specification version 1.1 and version 1.2 were released. In

Figure 1 History of Bluetooth technology



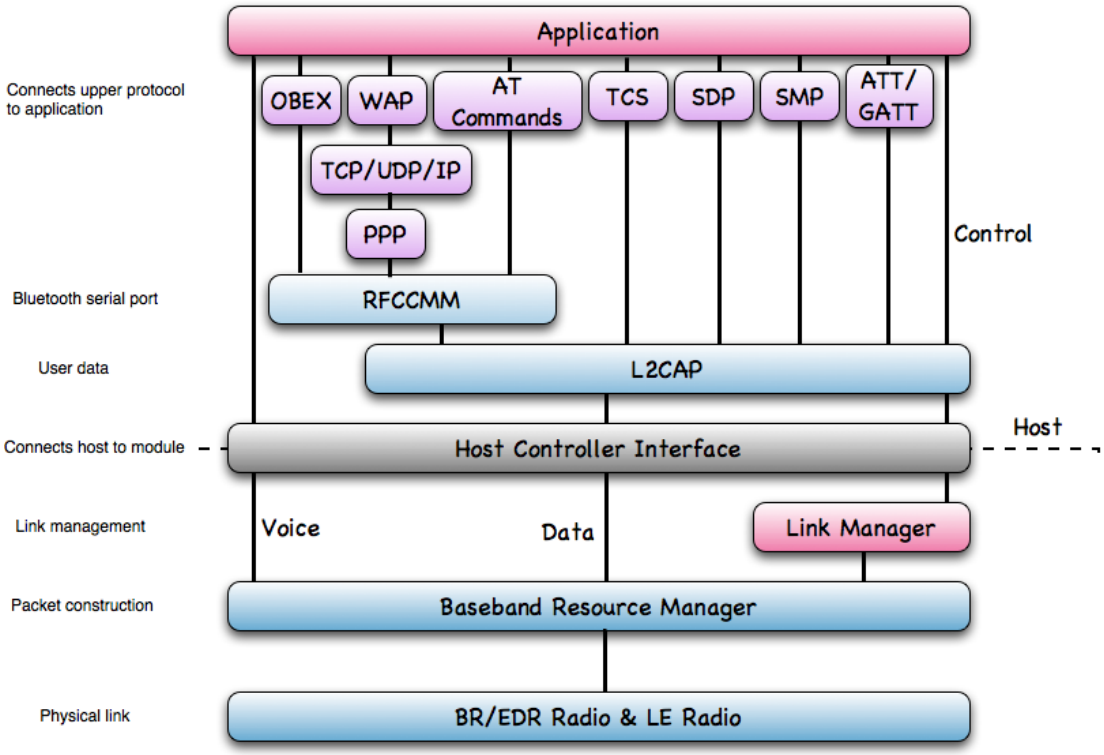
June 2004 and July 2007, Bluetooth core specification version 2.0 and version 2.1 were adopted by the Bluetooth SIG. The main difference of these two versions is that version 2.1 provides Enhanced Data Rate (EDR) as optional feature for faster data transfer. In April 2009, the Bluetooth SIG adopted the Bluetooth core specification version 3.0 plus High-Speed (HS), same as EDR, HS is also an optional feature. The latest version of Bluetooth specification is version 4.0 released in 2010, with the feature of Bluetooth low energy technology.

3.3 Bluetooth protocol stack

The structure of Bluetooth core system architecture can be divided into three parts: low-level hardware module, middle-level protocols, higher application level. Bluetooth technology standard is for the inter-operation between various applications. Remote devices should use the same protocol stack to inter-operate. In Bluetooth system, in order to support different applications, several protocols are required, and those protocols are organized in a hierarchical structure, called Bluetooth protocol stack, it is the core part of Bluetooth technology. Figure 2 shows the Bluetooth protocol stack of Bluetooth core specification version 4.0.

The low-level hardware module is the basis of Bluetooth system architecture, it includes Radio Frequency (RF), baseband, and Link Manager (LM). Bluetooth Basic Rate (BR)/Enhanced Data Rate (EDR) radio and Low Energy (LE) radio is the lowest layer. This layer works as transmitter and receiver, responsible for allocating radio band and channel. Baseband is between Bluetooth radio frequency and Bluetooth protocol stack, which together with radio frequency

Figure 2 Bluetooth protocol stack (Bluetooth core specification version 4.0)



layer constitute the physical layer. It includes tasks such as device discovery, link formation, synchronous and asynchronous communication with peers, define baseband packet format, flow control, forward error correcting and automatic repeat request. Link Manager is the data link layer protocol, takes care of link configuration, authentication, security, quality of service (QoS) (Ferro & Potorti, 2005).

The middle-level protocols provide services like QoS control, fragmentation and reassembly of data frame to upper layer applications. This level includes following protocols:

Host Controller Interface (HCI) protocol, which defines a standard interface-independent method to communicate with Bluetooth chip – unified interface and ways to access LM and baseband, the communication is done by using HCI commands, this enables the high degree portability of Bluetooth stack software (Bhagwat, 2001).

Logical Link Control and Adaptation Protocol (L2CAP), which is the core protocol in Bluetooth system, work in parallel with LM, provides data services to upper layers, protocol multiplexing, segmentation, reassembly, and QoS authentication. The L2CAP connection is established immediately when one Bluetooth device enter the radio proximity of the other Bluetooth device (Bhagwat, 2001).

Radio Frequency Communication (RFCOMM) is a protocol replace cables by creating virtual serial data stream, it supports binary data transport. To implement serial communication, it emulates RS-232 cable connection on top of L2CAP protocol (Wikipedia, 2011). Point-to-Point Protocol (PPP) is a data link protocol that establishes direct connection between two terminal devices, it is based on the interface of serial line, which is provided by RFCOMM. On top of PPP are Transmission Control Protocol (TCP), User Datagram Protocol (UDP) and Internet Protocol (IP), which use the packet-oriented interface of PPP. AT commands is a set of commands that control Bluetooth module on mobile phone, it also needs serial line interface support from RFCOMM to start an inquiry, built and disconnect a connection, and pair devices.

There are other middle level protocols, which are on top of L2CAP, like Telephony Control Protocol-binary (TCS), which defines the call control signaling of establish a voice or data call between Bluetooth devices (Wikipedia, 2011). Service Discovery Protocol (SDP), which is of vital importance in Bluetooth system architecture. Any Bluetooth application requires some certain services, and SDP defines the dynamic means to discover services that are provided by other Bluetooth devices, as well as the attributes of the discovered services. Then, SDP will build a communication channel for the service. Security Manager Protocol (SMP) is a peer-to-peer protocol used only in Bluetooth Low-Energy (LE) systems (e.g. in Bluetooth core specification version 4.0) to generate and management encryption and identity keys (Bluetooth Org, 2010).

Higher application level protocols include Object Exchange Protocol (OBEX), which transfer files between Bluetooth devices; Wireless Application Protocol (WAP), which is an open standard to provide the access to telephony and information services for mobile users (Stallings, 2005); Attribute Protocol (ATT) and Generic Attribute Profile (GATT), which are used for discovering services, and they are mandatory in a LE systems (Bluetooth Org, 2010).

3.4 Linux Bluetooth protocol stack

OpenBT is the first open source Bluetooth stack, it was developed in 1999. Then, in 2000, IBM developed another open source Bluetooth stack called BlueDrekar, but it was expired in 2002. Currently, there are two widespread Linux Bluetooth stacks, BlueZ and Affix. Affix is developed by Nokia Research Center, it supports core Bluetooth protocols, including HCI, L2CAP, RFCOMM, SDP, and various Bluetooth profiles (e.g. General Access Profile, Service Discovery Profile, Serial Port Profile, OBEX File Transfer Profile) (Affix, 2011).

BlueZ is the official Bluetooth protocol stack in Linux, it was initially developed by Qualcomm in 2001. BlueZ supports Bluetooth core layers and protocols, it consists of many separate modules, such as Bluetooth kernel subsystem core, L2CAP kernel layers, RFCOMM, HCI, general Bluetooth and SDP libraries (BlueZ Org, 2011).

In this study, we use BlueZ, it has been included in Linux kernel since version 2.4.6. Both the Bluetooth capability of Maemo OS and Android OS are based on BlueZ (Maemo Org, 2010; Android Developers, 2011). BlueZ uses D-Bus interface to hide hardware details, it is real hardware abstraction. It provides a set of APIs to facilitate developers to communicate with Kernel level protocols. Besides, it supports multi-threaded data processing as well as multiple Bluetooth devices (BlueZ Org, 2011).

3.5 Bluetooth technology usages

Various Bluetooth profiles support many different kinds of Bluetooth usage models, following are some usage models defined in Bluetooth specification (Bluetooth Org, 2010; Northstream, 2001; Rathi 2000; Kardach, 2000): Cable replacement, it is the initial purpose of Bluetooth technology. Quite many Bluetooth applications are based on this usage model, which enables the wireless communication between devices. Ad hoc networking, like exchange files between Bluetooth devices; Internet bridge, which provides Dial-Up Networking capabilities to PC by using wireless connection with mobile phone or cordless modem; Synchronization, it refers to data synchronization between devices.

Initially, the application of Bluetooth technology is limited in cable replacement, and now it is in widespread use. It has been applied in many dominant consumer products, such as adapters, mice, keyboards, and headsets. Especially in mobile phone, Bluetooth has been the standard. Besides these, MP3, MP4, digital photo frame, gamepad, stereo, and Skype phone also begin to jump on board. Moreover, the usage of Bluetooth technology is shifting to other fields, like home entertainment, vehicle applications.

4 Selection of targeted platforms and development tools

We use constructive research method in this thesis, aim to develop a measurement system to solve our research problems. This chapter discusses the selection of targeted platforms and development tools. The targeted devices used to run the measurement system have been determined as portable mobile devices, such as mini-laptop and mobile phones, which are small in size. First three sections present the targeted platforms – Ubuntu Netbook Remix, Maemo, Android, and then we introduce the development tools – Qt, Eclipse.

4.1 Ubuntu Netbook Remix

The first targeted device is Asus EEE PC 900, it runs a Linux OS called Xandros, but instead, we install another version of Linux OS – Ubuntu Netbook Edition for it, which is tailored to low capacity netbooks with small screens.

Ubuntu Netbook Remix is a free and open source software that belongs to the Unix-like OS family. The main feature of this version of Linux OS is that it uses GNOME and Unity user interface, it is designed specifically for netbook's small screen and for computing on the move, it provides an "Easy-mode" interface, and with the tabbed screen it is easy to find your preferred applications while other applications are still active on your screen (Ubuntu, 2011).

Minimum system requirement is 1.6GHz Intel Atom processor for installing Ubuntu Netbook Remix, and 1024 MB Random-access memory (RAM) as well as 4GB Flash disk or hard disk is recommended (Ubuntu Documentation, 2011).

There are several ways to install the Ubuntu Netbook Edition OS, downloading directly from Ubuntu server or using Wubi installer, in this study, we install Ubuntu Netbook Edition repository and other relevant packages after we install a regular Ubuntu package. Albeit the Asus EEE PC 900 has only 900 MHz processor, the Ubuntu Netbook Remix works well on it, and is sufficient for the research needs.

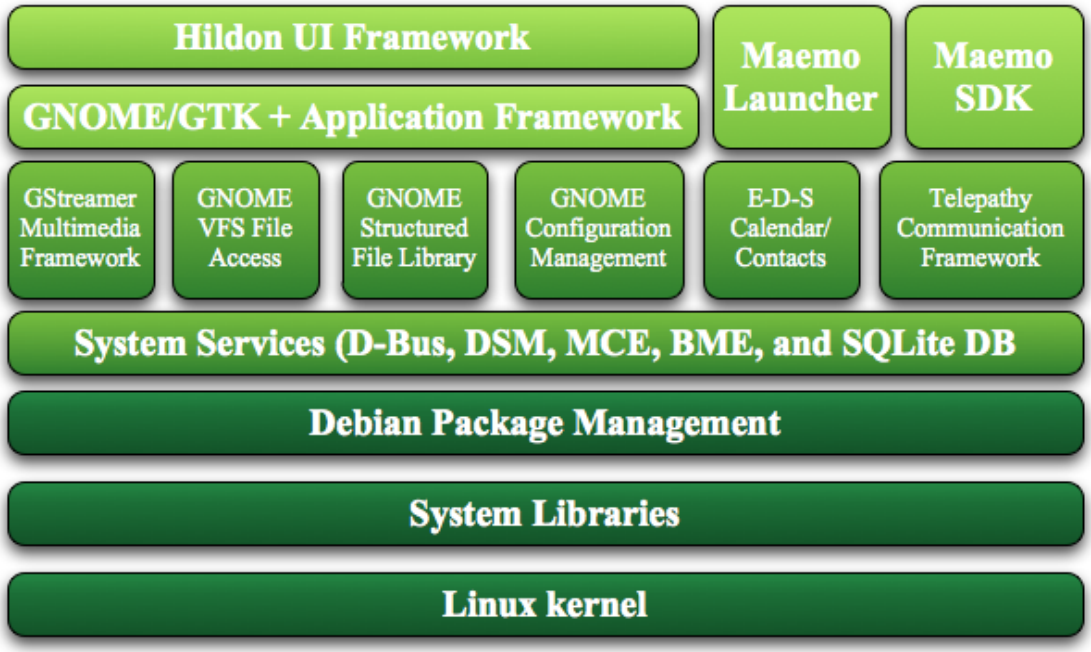
4.2 Maemo 5 mobile operating system

The second targeted device is Nokia N900, which has Maemo 5 as its default operating system. Maemo platform is based on Linux operating system kernel and many other open source components, such as Debian GNU and GNOME. Figure 3 shows the architecture of the Maemo platform.

Linux kernel is the central component that provides hardware layer; System Libraries includes GNU C library that is the basis of Maemo platform, OpenSSL for network security, and libcurl for accessing HTTP; Debian Package Management is the file system hierarchy; System Services provide services to applications and end users; Higher-level libraries, for instance GNOME Structured File Library; Maemo Launcher is responsible for launching applications on the platform; the user interface of Maemo is based on GNOME/GTK +, and Hildon user interface Framework support control panel, status bar, task navigator, and home applets (Maemo Org, 2011).

At first, we choose C++ programming language and Qt development tool for developing the measurement system, and Maemo 5 can support C++ language besides Python and C. In addition, Maemo 5 brings in Qt libraries, these Qt libraries will take care of giving the Hildon lock & Feel and enabling the virtual input methods for applications (Maemo Org, 2010), so we can easily port the Qt desktop application that runs on Asus EEE PC 900 to Maemo 5 platform.

Figure 3 The architecture of Maemo



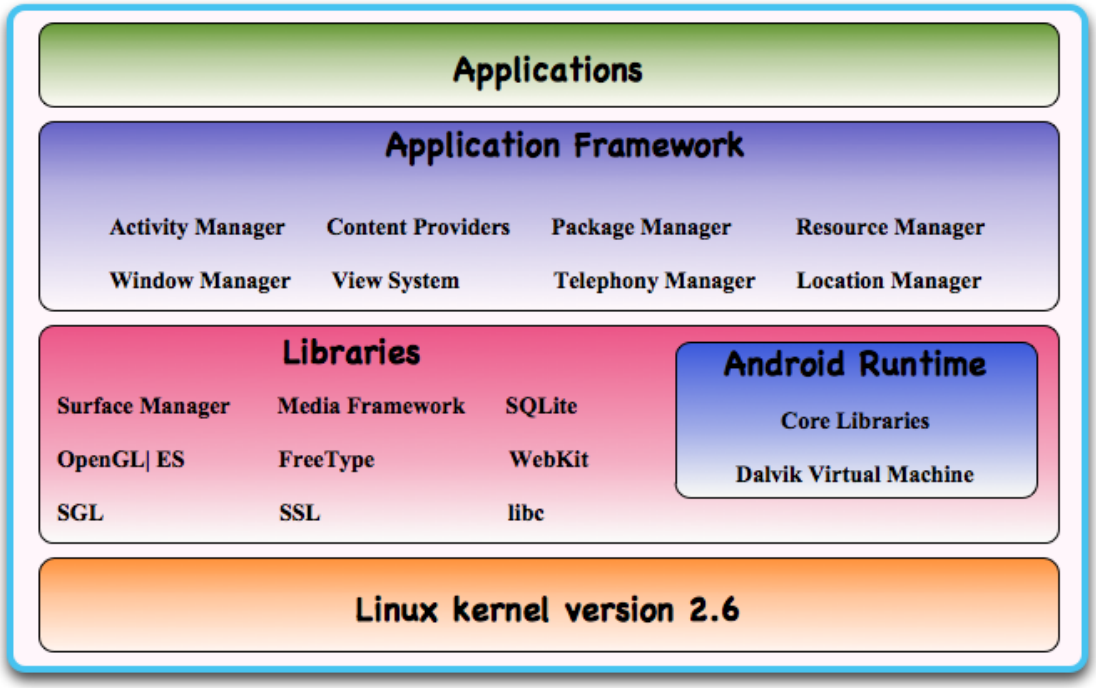
4.3 Android 2.1 operating system

After we developed measurement systems for both Ubuntu and Maemo platforms, a third mobile operating system is chosen – Android platform. We use HTC Wildfire mobile phone in our research, it runs Android 2.1. Android is also an open source mobile operating system based on Linux kernel version 2.6, the development and release are taken in charge by the Open Handset Alliance. The Open Handset Alliance includes a large group of companies, such as mobile operators, handset manufactures, semiconductor companies, software companies, and commercialization companies.

The Android platform has several features, it is an open source platform, it has robust operating system kernel and multiple open-source libraries, and Dalvik virtual machine that is optimized for mobile devices. It has a variety of tools to support development (e.g. Android Development Tools plug-in for Eclipse IDE, device emulator, JDWP debugger). Figure 4 shows the architecture of Android platform, the kernel is the abstraction of hardware layer, Libraries include a set of C/C++ libraries used by Android platform components, Application Framework layer provides services to applications (Android Developers, 2011).

It is not possible to directly port GNU/Linux or Qt applications to Android platform, and most of the Android applications are written in Java, so in the study, we rewritten the measurement system in Java for Android platform.

Figure 4 The architecture of Android



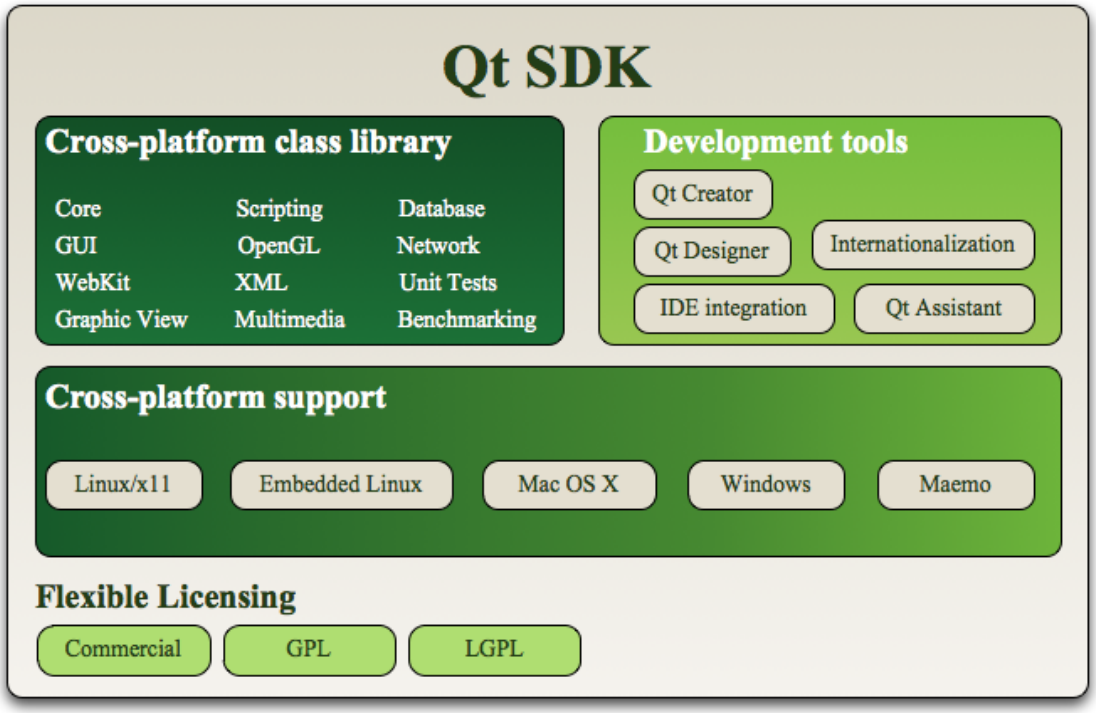
4.4 Qt Creator development tool

Qt is chosen as the development tool for Ubuntu and Maemo OS. Qt is a cross-platform application and graphical user interface framework, it is platform independent, and can support Linux/X11, Embedded Linux, Mac OS X, Windows, Symbian, and Maemo. It has the feature of develop once and portable across these platforms. It includes a cross-platform class library, several development tools (e.g. Qt Designer provides drag & drop user interface creation, Qt IDE Integration add-in for Eclipse and Visual Studio, Qt Internationalization, Qt help system), and the cross-platform integrated development environment (IDE) – Qt Creator. There are three Qt licensing options: Qt Commercial Developer License, with which developer can develop proprietary and/or commercial software; Qt GNU LGPL version 2.1, for developing proprietary or open source applications; and Qt GNU GPL version 3.0, for developing open source applications.

Qt Creator provides C++ and JavaScript code editor, Integrated user interface designer, project and build management tools, debugger plug-ins (e.g. GNU Symbolic Debugger (gdb), Microsoft Console Debugger (CDB), internal JavaScript debugger), version control systems (e.g. Git, Subversion, Perforce, CVS and Mercurial), simulator for mobile targets (Qt, 2011). Figure 5 shows Qt SDK.

Qt is very well documented, in this study, we use Qt Creator to develop measurement system for Linux operating system, and use Qt Creator combined with Nokia Qt SDK to develop mobile application for Maemo OS.

Figure 5 The architecture of Qt



4.5 Eclipse development tool with ADT plug-in

Qt is written in C++, but for the Android platform, applications are written in Java programming language. So another development tool Eclipse as well as ADT plug-in are chosen. Eclipse is a multi-language IDE whose ability can be extended by those Eclipse plug-ins. And ADT is one of Eclipse plug-ins, which supports for building Android targeted projects. Eclipse with ADT is highly recommended by Android Developers, ADT helps create application user interfaces, adds components based on the Android Framework APIs, and provides Android SDK tools for debugging (Android Developers, 2011). The most convenient thing is that it's very easy for developers to export signed/unsigned Android Package (APK) by using ADT, especially when compared to making a Debian package.

5 Measurement system

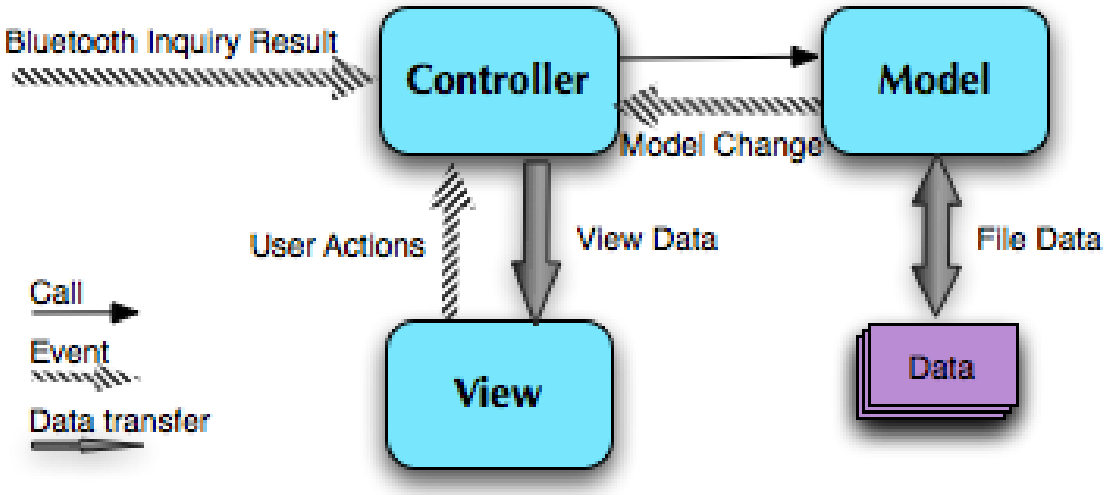
This chapter depicts the design of the measurement system in detail. First, it illustrates the architecture of the measurement system and the development process. Then, it describes the system requirements analysis by using Use Case diagram, and discusses the structure and detailed logic of the measurement system by using Class diagram, Activity diagram and Sequence diagram. Next, it presents the design of graphical user interface. Last, it discusses the developed three prototypes, and compares the differences of Bluetooth APIs in the three platforms.

5.1 Architecture of measure system

Model-View-Control (MVC) first was a software design pattern designed for the Smalltalk-80 programming language by Xerox PARC in the 80s. In recent years, it has been widely used as software architecture pattern to describe the structure of an application program, the responsibilities and interaction ways of modules.

Contemplating the background of VESC project, the targeted terminal devices that are used to run the measurement system should be portable and small-size, for example, mini-laptop and mobile phones. This brings forward challenges for supporting different types of devices, so we applied MVC to the development of our measurement system. The most dramatic advantage is that the presentation of user interface is clearly separated from the application logic. This facilitates the testing with the measurement system, and it also enables code reuse when system is ported to another platform. Figure 6 shows the MVC architecture of the measurement system.

Figure 6 MVC architecture of measurement system

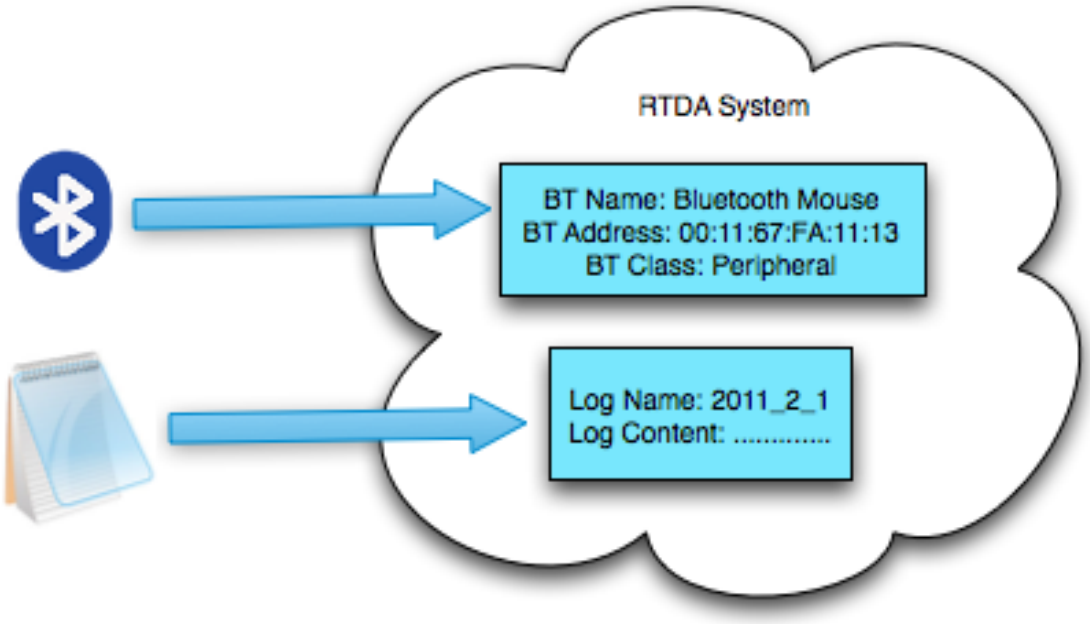


The idea behind MVC architecture pattern is very simple, the application program has to distinguish the following responsibilities:

Model refers to the business data and business logic, is the main part of the application. In software systems, everything can be abstracted into data model, which can be processed in certain ways. For example, in Figure 7, the Bluetooth devices and text file logs are data that have to be processed according to corresponding rules. In the measurement system, the model is responsible for detecting Bluetooth device, managing allowed Bluetooth devices, recording real-time data, and managing file I/O.

View is the interface that users can see and interact with. One model can have several views to present model data in different formats and ways, and it is the controller makes the selection of views. In the measurement system, view visualizes the results of real-time data acquisition, and provides user interaction.

Figure 7 Data model example



Controller executes user requests by calling objects and coordinating needed resources. Controller itself doesn't do processing, or output any results, it calls the corresponding model to handle it, and then calls the view to present the view data return from the model. In the measurement system, controller takes care of the Bluetooth device inquiry responses, receive user actions, call model to execute user request, receive model changes, update views.

5.2 Development process

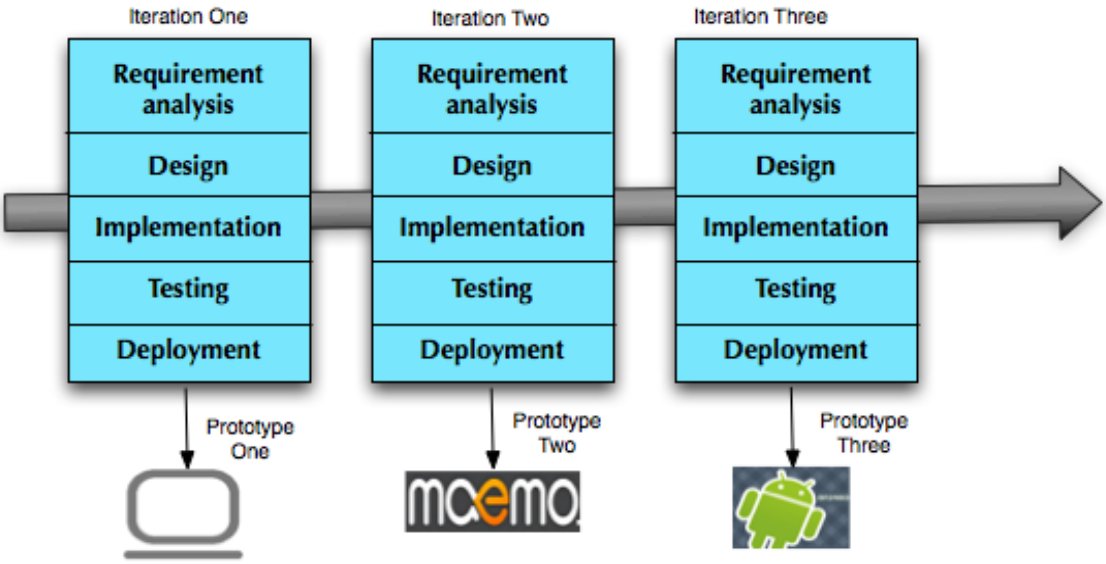
In the implementation of the measurement system, we plan to use different types of Linux or Linux-based devices to run the system, which collects real-time process data by detecting currently available Bluetooth devices. We use following three Linux-based platforms:

- Prototype one: Asus EEE PC 900 with Ubuntu OS
- Prototype two: Nokia N900 with Maemo 5 mobile OS
- Prototype three: HTC Wildfire with Android 2.1 mobile OS

We use software prototyping approach combined with iterative software development process in the study due to the difficulties to identify requirements completely at the beginning phase of the development of the measurement system. Rational Unified Process (RUP) and eXtreme Programming (XP) both recommend the adoption of iterative model, it makes adjust to constantly changing requirements easier, it is a more flexible and less risky method, it is suitable to quickly start the development of the measurement system.

One iteration is a complete development cycle, including a sequential software development processes: requirement analysis, design and implementation, testing and evaluation. The basic

Figure 8 Prototyping & iterative development process



idea of iterative model is that each iteration generates one executable version of product, from iteration to iteration, incrementally developed subsets constitute the final production at the end. Prototypes of the measurement system are created in each iteration, as shown in Figure 8.

In iteration one, the first prototype is developed for mini-laptop that runs Ubuntu OS, and a laboratory testing is conducted to evaluate it. Second iteration transfers the prototype one to Maemo platform. Prototype two is tested in a practical testing environment in Helsinki University Central Hospital. Prototyping is helpful for understanding the system, and with the iterative model, useful information can be learned by using and evaluating prototypes repetitively. The third iteration takes advantages of the analysis results of the development of previous two prototypes, Android platform is chosen, and new requirements are added into the development of prototype three.

Same as Agile software development method, iterative model welcome changing requirements. In addition, testing in each iteration helps detect bugs and errors and corrects them in early stage, this enables the robustness and high quality of the system.

5.3 Measurement system analysis

This section uses Unified Modeling Language (UML) to analyze and design the system. It gives comprehensive descriptions of the measurement system both from static and dynamic views.

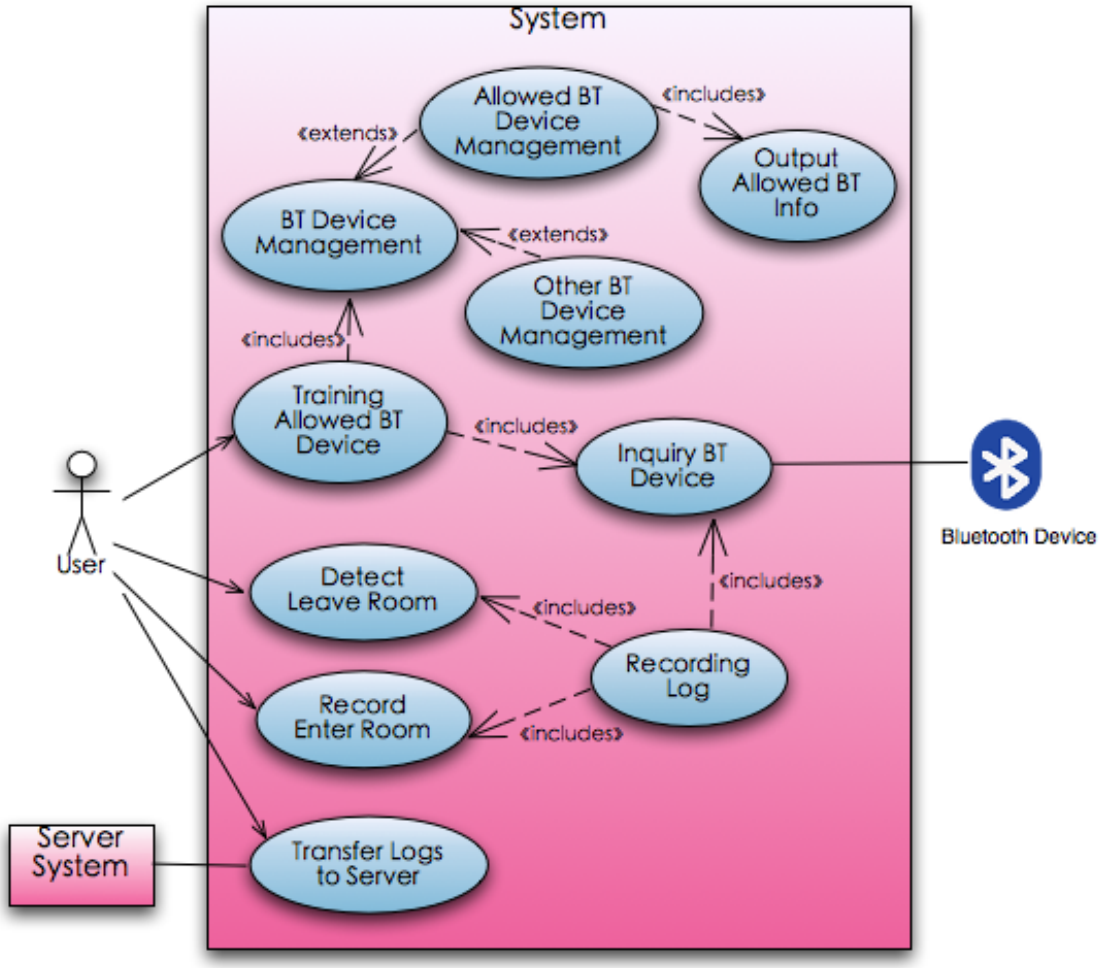
Use case diagram

Usually, use case diagram is used to determine the functional requirements of a system, one ellipse represents one use case. And there are three types of relationships between use cases – inheritance, includes and extends, two kinds of associations between actors and use cases – in-

volved in the use case and initially invoke a use case, and maybe inheritance relationship between actors.

Figure 9 shows the boundary of the system and its use cases. User is the person who interacts with the system, and the system will record his/her activities. Bluetooth devices are set at different places to represent different activities, once the system enters the radio proximity of a remote Bluetooth device, a wireless ad hoc connection is established between them [7], the system sends out discovering request and the remote Bluetooth devices will response the request by sending back relevant information (e.g. Bluetooth address, user-friendly name, device type, connection signal strength). The system includes three main use cases. First, an infinite loop of inquiring remote Bluetooth devices and process responses (see Inquiry BT Device use case in Figure 9), it is the fundamental of the system. Second, Recording Log use case in Figure 9, which collects real-time data, determines activity's timestamps, and creates accurate event logs according to the needs of automatic process measurement. Third, Training Allowed BT Device use case, it means to train the system to recognize which Bluetooth devices are allowed. The third use case is developed in iteration three, based on the flexibility problems found in iteration one and two.

Figure 9 Measurement system requirement analysis by using use case diagram



Following tables describe some of the use cases in detail. Table 2 delineates the use case that inquiry remote Bluetooth devices, Table 3 presents training allowed devices use case, which training the system recognizes allowed Bluetooth devices and Table 4 shows recording logs use case.

Table 2 Inquiry remote Bluetooth devices use case

Use Case Name	*Inquiry BT Device*
Use Case No	5,1
Primary Actor	Bluetooth devices
Description	Inquiry remote Bluetooth devices and process responses
Precondition	Local Bluetooth adapter exist, and is enabled
Basic Flow	– The use case begins when user open the application – Receive responses from remote Bluetooth devices – Use Case: BT Device Management is performed – The use case will be called when one inquiry is finished
Alternate Flows	Local Bluetooth adapter is not available

Table 3 Training allowed Bluetooth devices use case

Use Case Name	*Training Allowed BT Device*
Use Case No	5,2
Primary Actor	User
Description	Training the system to recognize those allowed Bluetooth devices
Precondition	Local Bluetooth adapter exist, and is enabled
Basic Flow	– The use case begins when user start training Bluetooth devices – Use Case: Inquiry BT Device is performed – Use Case: Other BT Device Management is performed – Use Case: Allowed BT Device Management is performed – User stop training – Use Case: Output Allowed BT Device is performed
Post-conditions	Allowed Bluetooth devices list have been saved in .txt file

Table 4 Recording logs use case

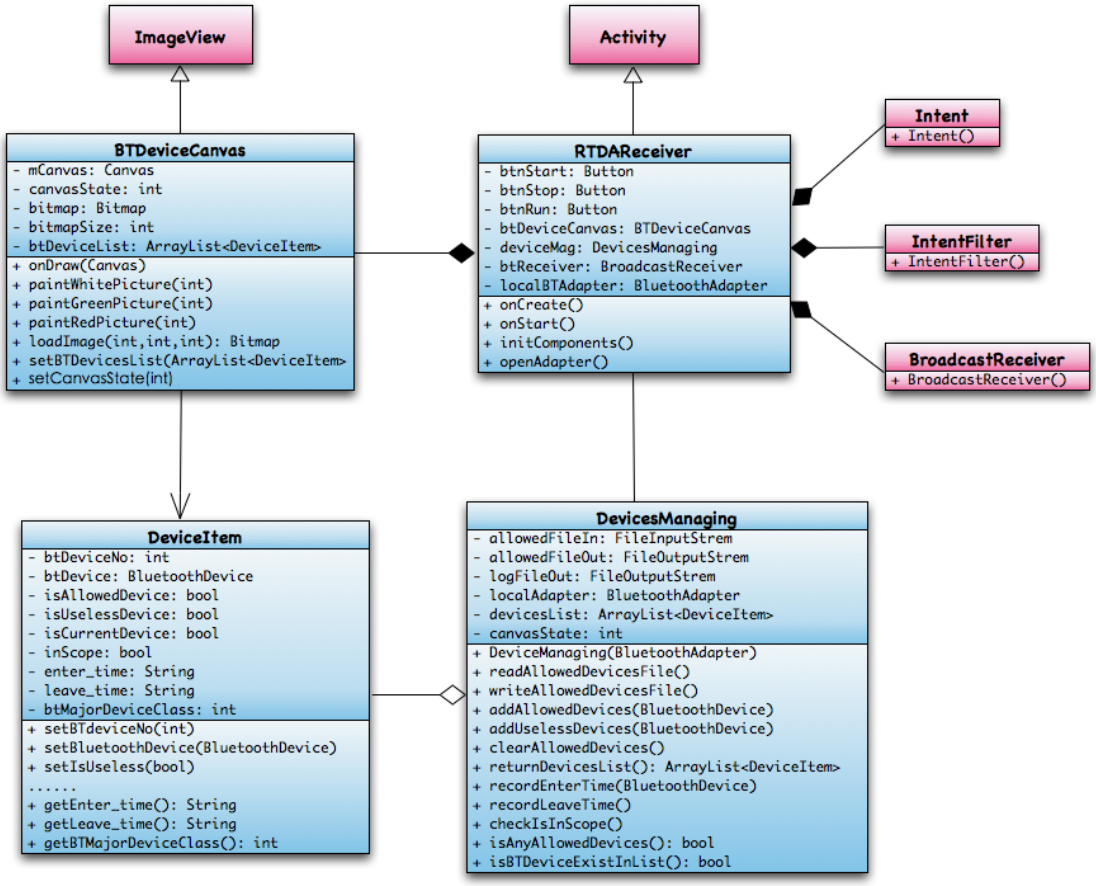
Use Case Name	*Recording Log*
Use Case No	5,3
Primary Actor	User
Description	Record activities in logs
Precondition	– Local Bluetooth adapter exist, and is enabled – Allowed Bluetooth devices is not null
Basic Flow	– The use case begins when user start recording activities – Use Case: Inquiry BT Device is performed – User enter a room that has allowed Bluetooth device inside – Use Case: Record Enter Room is performed – User leave the room – Use Case: Detect Leave Room is performed – The use case ends when user turn the application off
Post-conditions	Activity logs have been saved in .txt files

Class diagram

We use class diagram to describe the static structure of the system. Class diagram shows system classes, their attributes and operations, as well as the interrelationships between classes – association, aggregation/composition, dependency, generalization. Figure 10 shows the architecture of the third prototype, which has been developed and refined to be more suited to mobile devices, it consists of four components, which are RTDAReceiver, BTDeviceCanvas, DevicesManaging and DeviceItem.

RTDAReceiver inherits from Activity class, runs in application's main thread to create a window that can present application's interface, also responsible for interacting with users, managing local Bluetooth adapter and inquiring remote Bluetooth devices. As an Activity class, it has its life-cycle: onCreate() – first called to do static set up when the activity is created, onStart() – called after the activity is visible for user, onResume() – called when user start to interact with the activity, onStop() – called when the activity is invisible for user, onDestroy() – the last call before the activity is destroyed (Android Developers, 2011). In the measurement system, it has only one Activity class, means the RTDAReceiver Activity is always active and in the foreground of the screen.

Figure 10 The structure of the third prototype of measurement system



RTDAReceiver class is aggregated by Intent class, IntentFilter class, BroadcastReceiver class, and a self-defined ImageView class. Intent refers to the action to be performed, it acts as media typically: provides information for components calling each other, implements the decoupling between the caller and callee components. With IntentFilter class, the system can register actions dynamically, and only listen to interested actions, like 'ACTION_FOUND' and 'ACTION_DISCOVERY_FINISHED' actions. BTDeviceCanvas is a self-defined ImageView, it visualizes system's inquiring results.

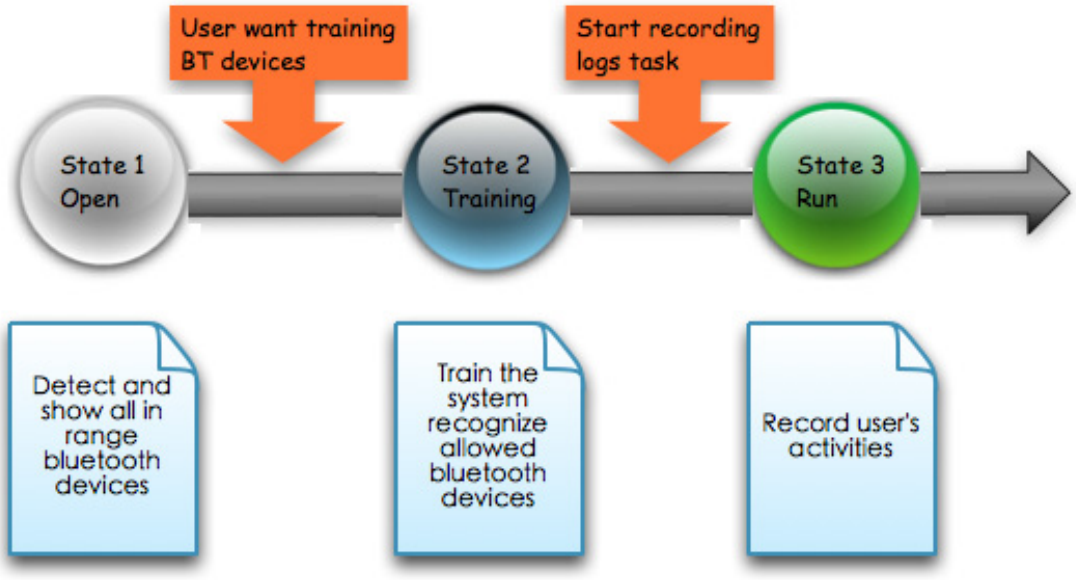
DevicesManaging manages allowed Bluetooth devices, takes care of file I/O management, records relevant information of activities, including information of remote Bluetooth device, activity's begin time and end time. It has a DeviceItem type list, which saves Bluetooth related information (e.g. name, address, device class) and Boolean variables to help manage devices (e.g. is a device allowed or it is irrelevant, the number of allowed devices), and other variables to support the task of recording logs (e.g. check if a Bluetooth device is still in range, the enter time and the leave time). In the first two prototypes, we use several lists to put different information, then we found a lot of work have to be done in terms of comparing lists. Moreover, it consumes more memory and makes device management and Logs recording more difficult. So in the third iteration, all information has been integrated into one single class, and Bluetooth device related operations are more centralized.

Activity diagram

In prototype one and prototype two, the allowed devices list is predefined and written into a text file manually, it lacks flexibility, which generates new requirements to implement the function of training allowed devices dynamically.

Allowed devices list should be automatically generated by the system to adapt to various running environments. Thus, we design three states to distinguish three different sets of system tasks, as shown in Figure 11, state one begins when the system is running, state two is trig-

Figure 11 Three states in prototype three



gered when the user wants to train the system recognize allowed Bluetooth devices, state three is triggered when the user starts recording logs for his/her activities.

Following part of this section uses separate activity diagrams to depict the workflow processes in these three states. When first open the system, it is in the "OPEN STATE", in which the system will present all in range Bluetooth devices, as shown in Figure 12.

First, some preparation works have to be done. DiviceManaging class reads allowed devices information that is saved in a text file into an ArrayList. RTDAReceiver class gets default local Bluetooth adapter, which is essential because local Bluetooth adapter is the basis of any fundamental Bluetooth operations. If no Bluetooth adapter is available in the mobile device, the application will be ended, otherwise, RTDAReceiver class will enable the Bluetooth of the mobile device and start inquiring remote Bluetooth devices.

Then the RTDAReceiver Activity class will start an asynchronous call that inquiries remote Bluetooth devices, each inquiry lasts for about 12 seconds. Once a remote device is found, the BroadcastReceiver will get 'ACTION_FOUND' intent, and when the inquiry scan is time out, it will get 'ACTION_DISCOVERY_FINISHED' intent. Bluetooth inquiry is a heavyweight procedure, is not managed by Activity class, but run as a system service (Android Developers, 2011), so in the third prototype we designed that a new inquiry only started after existing inquiry is ended. In prototype one and prototype two, we use a new thread to perform infinite loop to search for remote devices. That causes system instability problems. And the asynchronous call in prototype three returns immediately, won't affect user interface rendering, but this also brings up one problem – the inquiry result is unpredictable then.

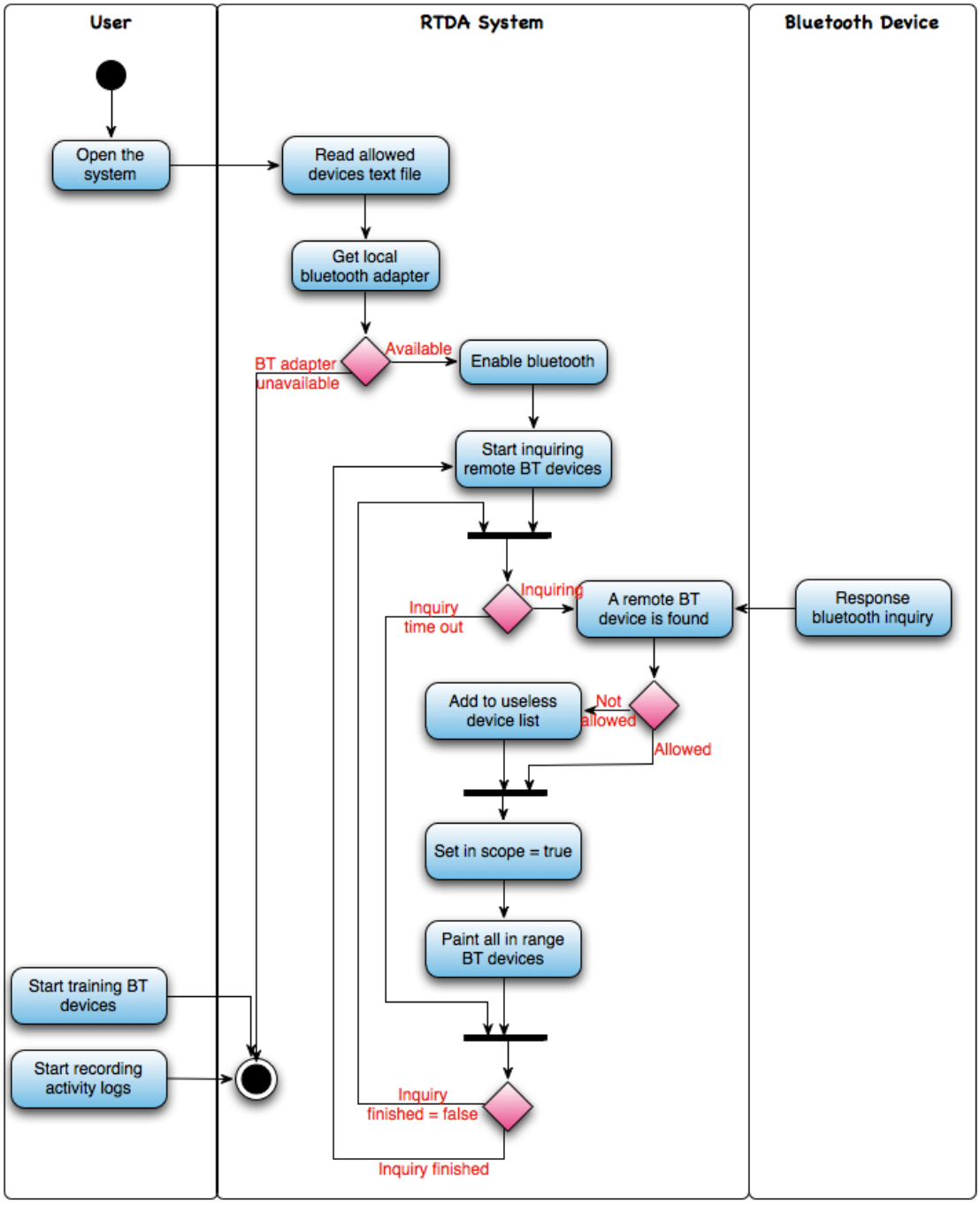
Three different system states adopt one common Bluetooth device inquiry process: start inquiry, find a remote device then do relevant tasks. If no Bluetooth device is in range, then wait for the scan becomes time out. When a discovery is finished, a new discovery will be started. According to the state of the system (described above in Figure 11), different tasks will be performed after the system catches 'ACTION_FOUND' and 'ACTION_DISCOVERY_FINISHED' intents.

In 'OPEN STATE', if detected remote devices are not in the allowed devices list, then it will be added into the irrelevant device list. The attribute 'inScope' of all detected devices will be set as true, and BTDeviceCanvas class will paint all those in range devices (including allowed and irrelevant devices).

Both when user starts training allowed devices and starts recording activity logs, the 'OPEN STATE' will end, and the system will transform to the other two states. There are two preconditions of training the system to recognize allowed devices. One, allowed Bluetooth devices can be enabled after the system transforms to 'TRAINING STATE' successfully. Two, the system should recognize all those irrelevant devices before the 'TRAINING STATE'. When the training is beginning, the system will remark detected devices as allowed. The result of training is a text file that saves information about these allowed devices. In this state, only allowed devices are shown to the users, Figure 13 shows the logic process of system in this state.

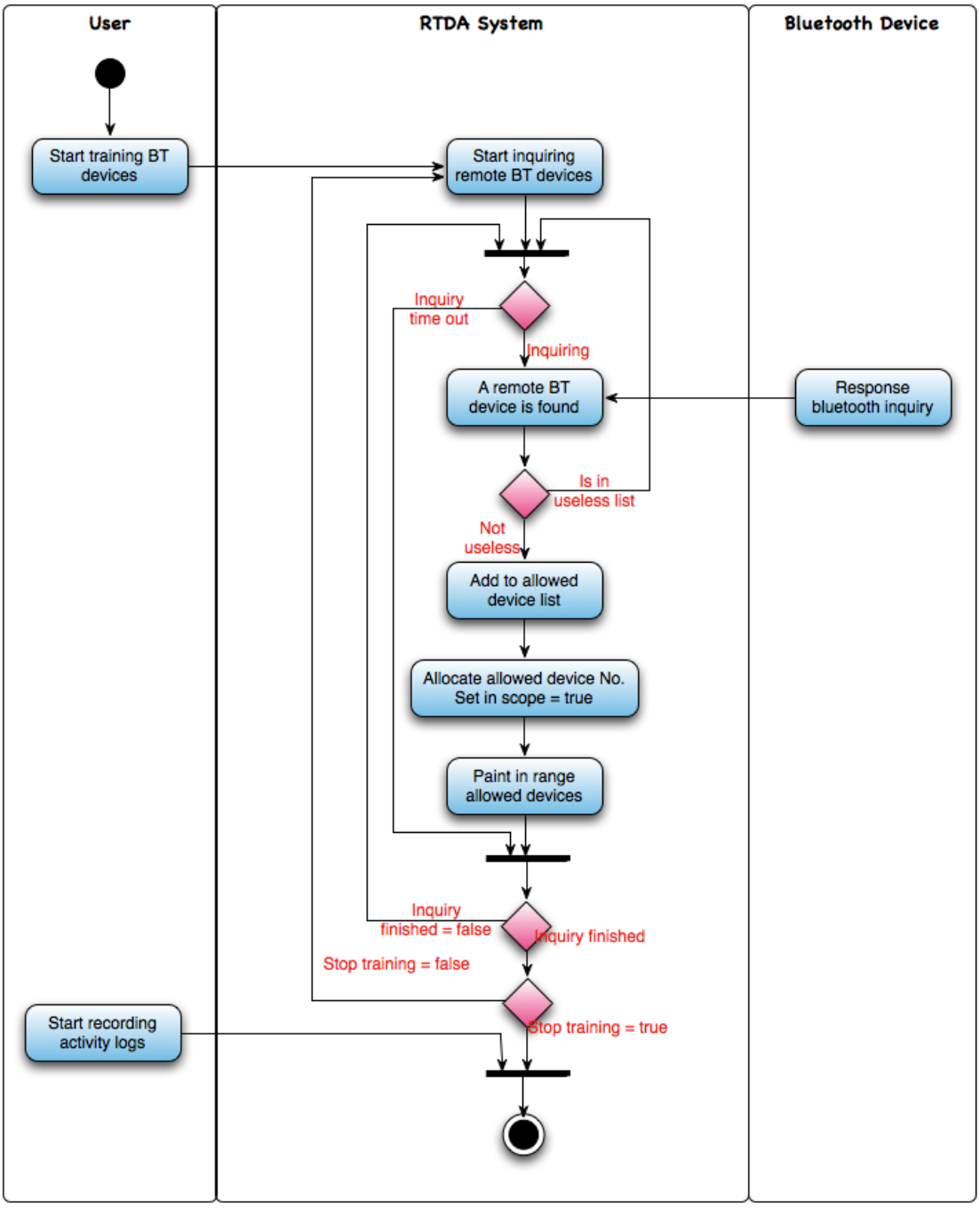
'RUN STATE' is the main part of the system, it records activity logs for further automatic process modeling use. Process modeling requires at least: activity ID, activity name, timestamps (begin time and end time), and user ID. The measurement system uses Bluetooth de-

Figure 12 The process of inquiring remote Bluetooth devices



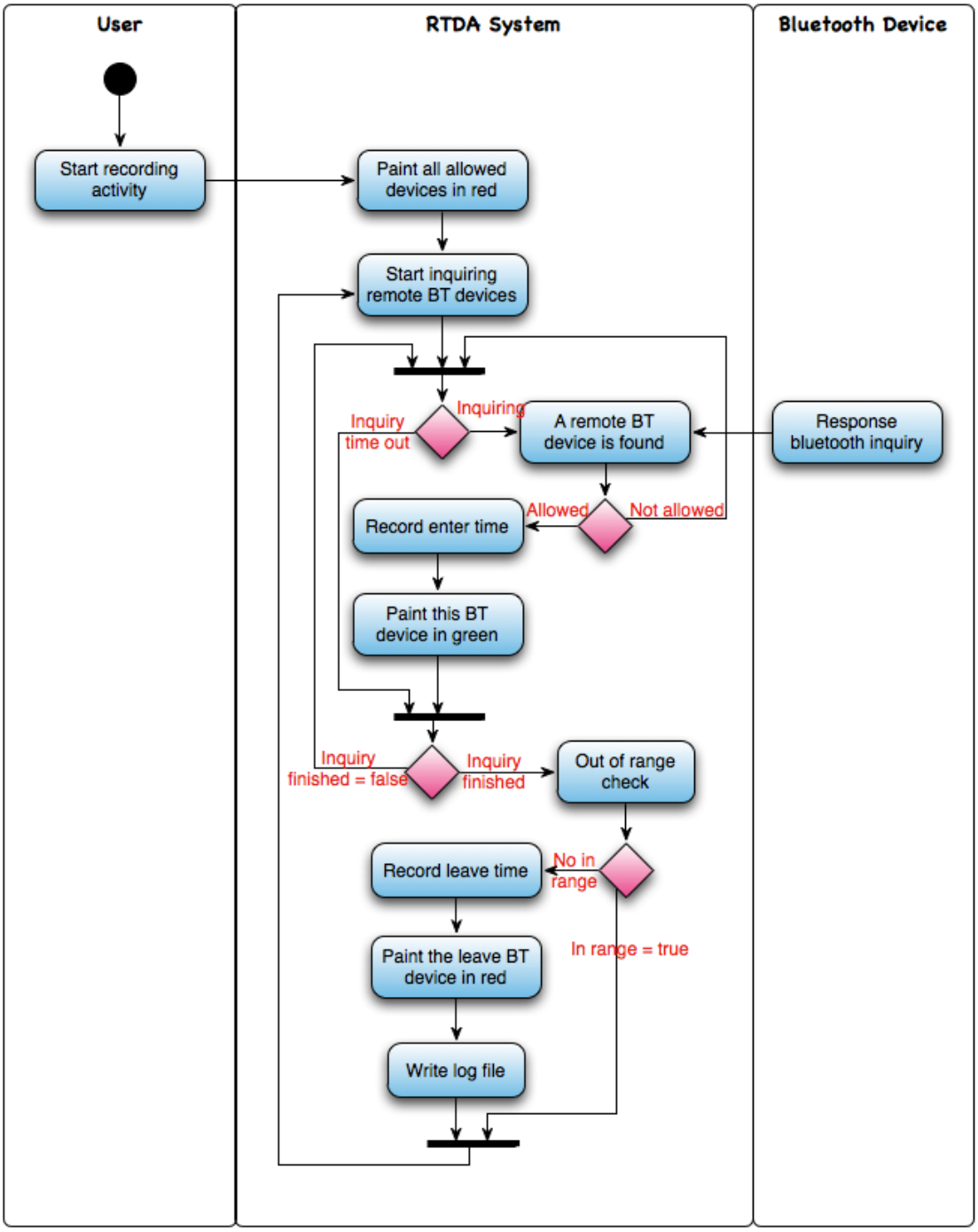
vice unique address to distinguish different activities. As for timestamps of an activity, we're facing some challenges, especially when detecting the leave time. It takes much longer to detect the leave time than detect the enter time. In prototype three, Bluetooth inquiry is an asynchronous call and 'ACTION_FOUND' intent is triggered when one remote device is found instead of return all in range Bluetooth devices' information once, so we use one solution to detect the leave time: use an extra attribute, this attribute will be set as 'true' when the Bluetooth

Figure 13 The process of training allowed devices (in training state)



signal of the device is detected by the system, and it will be set as 'false' when one discovery is finished. In next round discovery, if the same device can be found means it is still in scope, but if it cannot be found before the new round of inquiry is finished, then the attribute's value will keep 'false', it means the device isn't in range anymore, then we take this as the leave time. Compare to previous two prototypes, the accuracy of timestamps has been improved. And in Figure 14, it shows the workflow of system's main function.

Figure 14 The process of recording logs (in run state)

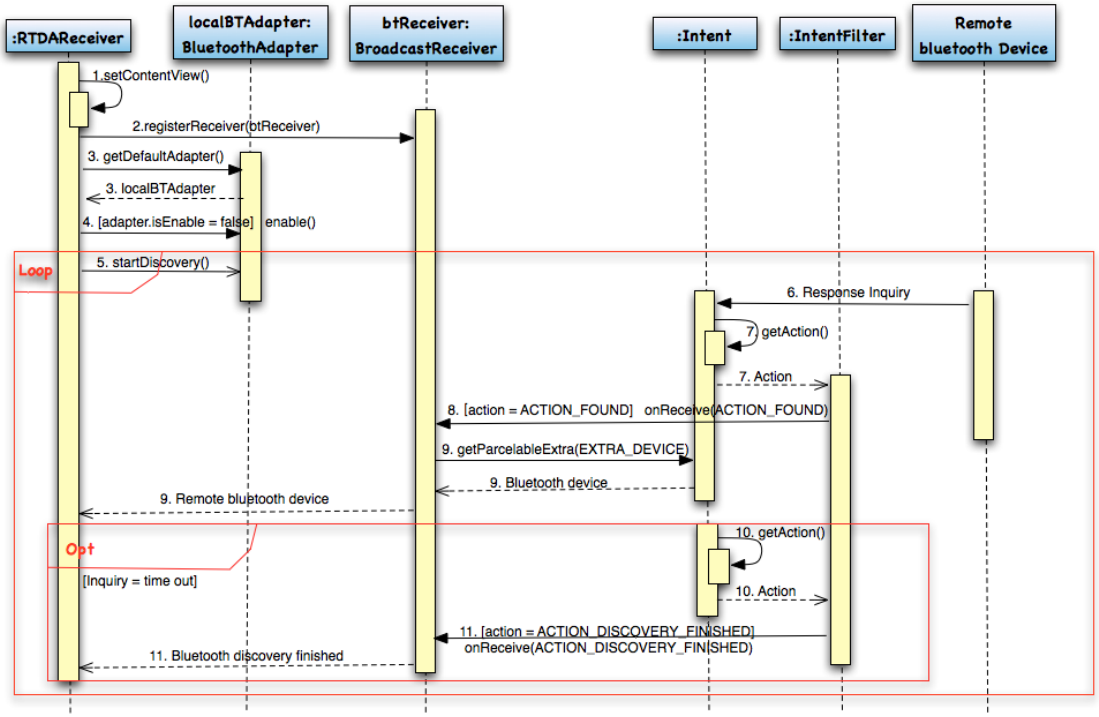


First, the system will present all allowed devices in red color, which means it isn't in range now. Then, it starts inquiring, once one remote device is detected, check if it is an allowed device, record the enter time of allowed devices and change those devices to green color, which means they are in range now. When one inquiry is finished, the system will check whether those Bluetooth devices have been detected in the earlier round still in range. If not, it records the leave time, turns the device into red color and writes this record into text log file.

Sequence diagram

To illustrate how those classes in the system interact with each other, we use sequence diagram. It focuses on the sequence of events, complements activity diagrams to present the system in another dynamic way, enables validate system's business process logic. Following are sequence diagrams for prototype three. Figure 15 shows the logic of Bluetooth devices inquiry loop, in different scenario, specific operations are performed inside this common inquiry loop after BroadcastReceiver returns 'a remote Bluetooth device is found' or 'Bluetooth discovery finished' message to the main Activity class, described later in Figure 16, Figure 17, Figure 18.

Figure 15 The logic of the loop of inquiring remote Bluetooth devices

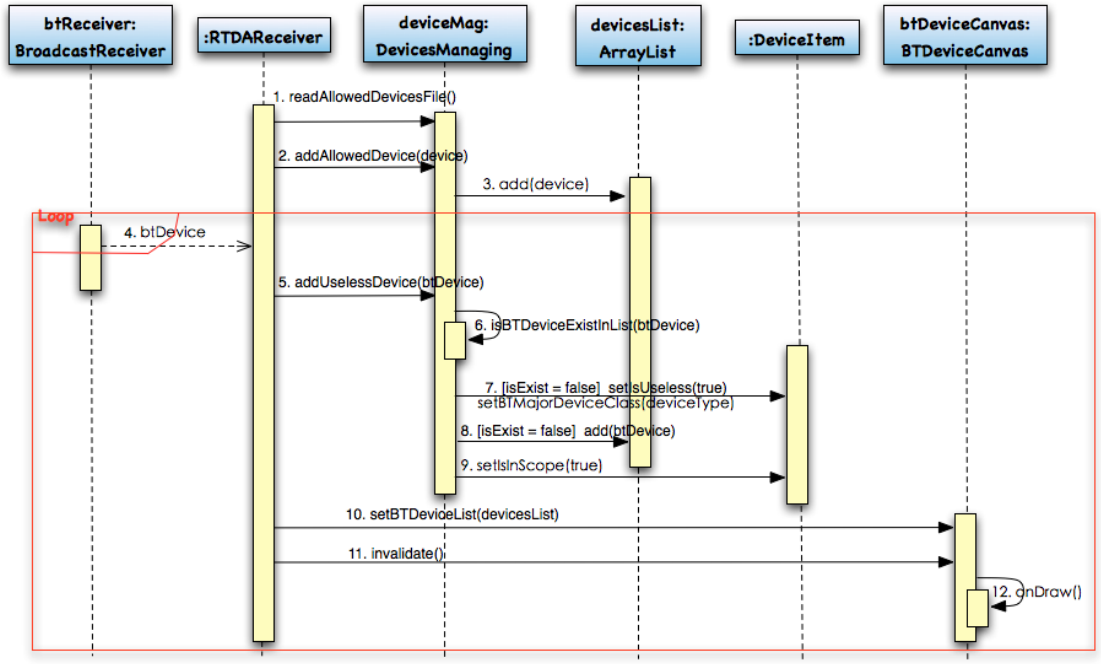


First, the RTDAReceiver Activity loads the user interface layout, calls setContentView() function, register 'ACTION_DISCOVERY_FINISHED' and 'ACTION_FOUND' intents. Others preparation works followed, gets default adapter, if the Bluetooth isn't enabled, enable() function will be called to enable the Bluetooth. Then, begins the inquiry loop, startDiscovery() is an asynchronous call, after call this function, the system can continue handle user interface tasks. When a remote Bluetooth is found, Intent class will get the action of the intent, if the action equals ACTION_FOUND, BroadcastReceiver's onReceive() function will be triggered, getParcelableExtra() is a function of Intent class, the system can get the remote Bluetooth device object by calling this function. One inquiry lasts for 12 seconds, events in the 'opt' will happen only when 'inquiry = time out'. A new round of loop will begin right after the discovery finished.

Figure 16 describes the scenario when the system is first started, what specific processes are conducted after BroadcastReceiver returns 'a remote Bluetooth device is found'.

The precondition of this scenario is the system reads information of allowed devices from the text file. DevicesManaging calls readAllowedDevicesFile() function, and uses FileInputStream class to read the file that store the information of allowed devices. Then, it calls addAllowed-Device() and ArrayList's add() functions to save it into an ArrayList for further device management.

Figure 16 The logic of the system when it is in open state



Following this is the specific processes in scenario 'OPEN STATE'. When a remote Bluetooth device responses to the inquiry and BroadcastReceiver catch the 'ACTION_FOUND' intent, RTDAReceiver will call addUselessDevice() function in DeviceManaging class. DeviceManaging class calls isBTDeviceExistInList() to check if this detected remote device already exist in the device list. If it exists, means it is allowed device, then sets isInScope equals true, this attribute is used when the self-defined ImageView class calls onDraw() function to check whether a device in the device list should be paint on the canvas. If the detected device doesn't exist, means it is an irrelevant device, it will also be added into the device list, but it will be marked as an irrelevant device by calling setIsUseless(true). Then, it calls setBTDeviceList() to pass a handle of device list to BTDeviceCanvas class, after invalidate the canvas, onDraw() function will be called to repaint the image, all in range devices will be painted on the canvas.

Figure 17 describes the scenario when user start training the allowed devices. OnClickListener catches the action of Start button is clicked, then BTDeviceCanvas will be informed that it is TRAINING STATE, and DeviceManaging class will clear all allowed devices exist in the device list. Next is the inquiry loop, similar as those processes described in Figure 16, the difference is that in this scenario, the system calls addAllowedDevice() function to mark detected devices as allowed devices, and only allowed devices will be painted. When OnClickListener is triggered by Stop button, DevicesManaging class will write the information of all newest detected allowed devices into a text file of allowed devices list.

Figure 17 The logic of training allowed devices

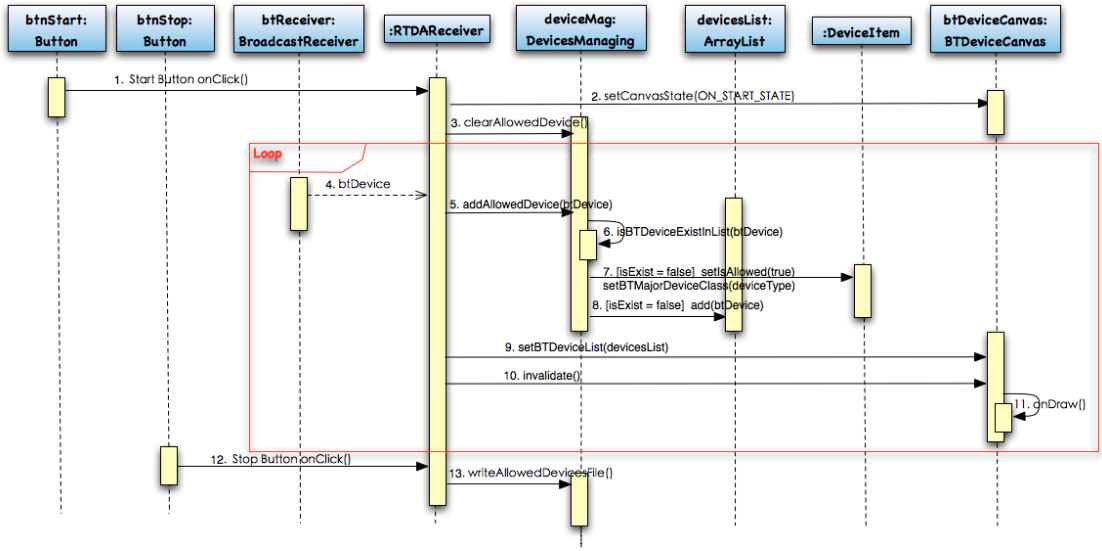
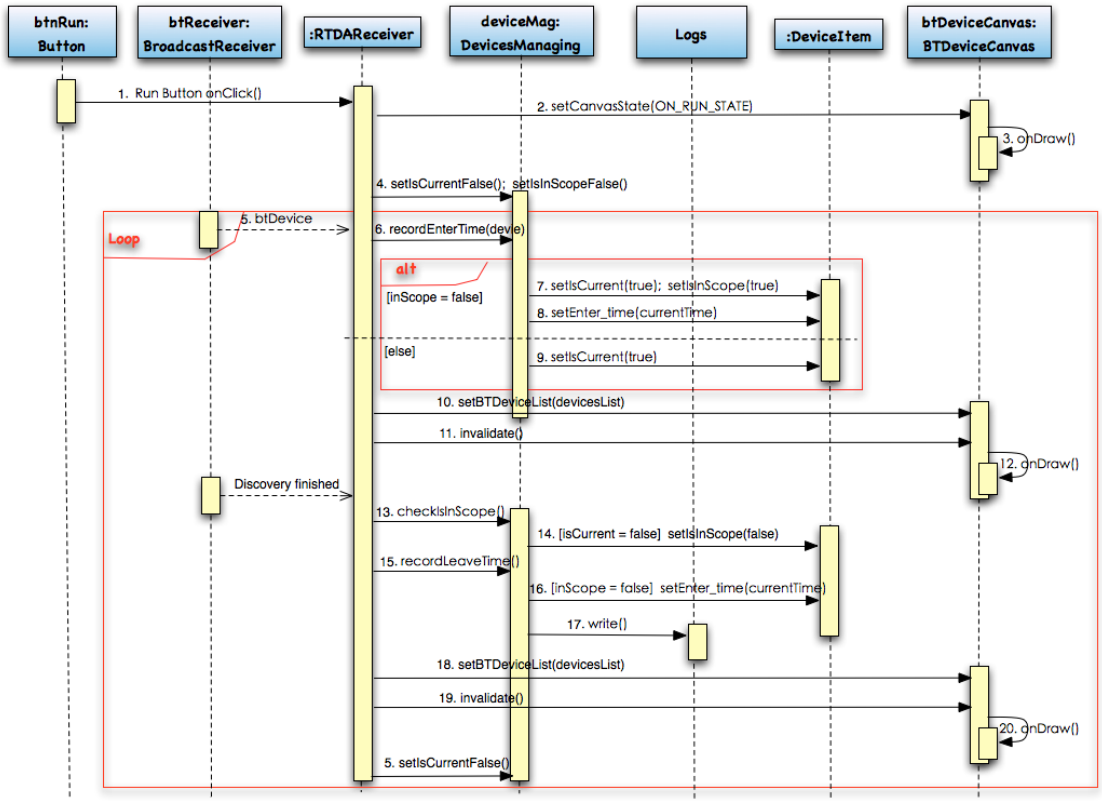


Figure 18 describes the scenario when user start recording their activities, same as illustrated in Activity Diagram in Figure 14. This sequence diagram focuses on the interaction between objects. Run button is clicked, BTDeviceCanvas will be informed that it is RUN STATE, and BTDeviceCanvas will repaint the image to show all allowed devices. Two Boolean attributes 'isCurrent' and 'isInScope' are used because the system needs an extra attribute 'isCurrent' to determine when one device is out of range.

First, setIsCurrentFalse() and setIsInScopeFalse() set all devices's isCurrent and isInScope attributes to false. When a remote device is detected by the system, recordEnterTime() function will be called, checks if the device isn't in range (isInScope = false), sets current time as the device's enter time, and sets isInScope and isCurrent equals true, then updates the user interface, paint this device is in range (use green color icon). If the device is already in range (isInScope = true), which means the user hasn't left the range of the device yet, so set isCurrent true.

The system will check the leave time only when one inquiry is finished, first calls checkIsInScope(), if one device's isCurrent = false, then sets its isInScope = false, this means the device hasn't been discovered before the 12 seconds' inquiry finished, so the device isn't in range anymore. Next, calls recordLeaveTime() function, if one device isn't in range (isInScope = false)

Figure 18 The logic of recording logs



and it has enter time, but without leave time, then sets current time as the leave time, uses FileOutputStream to write this piece of record into the log file, and repaints those leave devices with red color icons on the canvas.

Last, it sets all devices' isCurrent to false, but keep the value of isInScope. If the device can be detected in next round inquiry, isCurrent will be set to true, as described above. But, if the device cannot be discovered in the new round of inquiry, the isCurrent will keep false, and when the whole inquiry is finished, the system will determine this device as leave. We use two similar attributes to solve the problem: if the system changes the value of isInScope instead of using an extra Boolean attribute, all those in range devices will be set as leave first, and then set as enter, repeatedly, this makes one record be cut into many fragments.

5.4 User interface design

To better support the testing and measurement, we design simple and handy user interfaces to represent real-time workflows. Visualization enables direct observation of the measurement system's performance and real-time data acquisition's results.

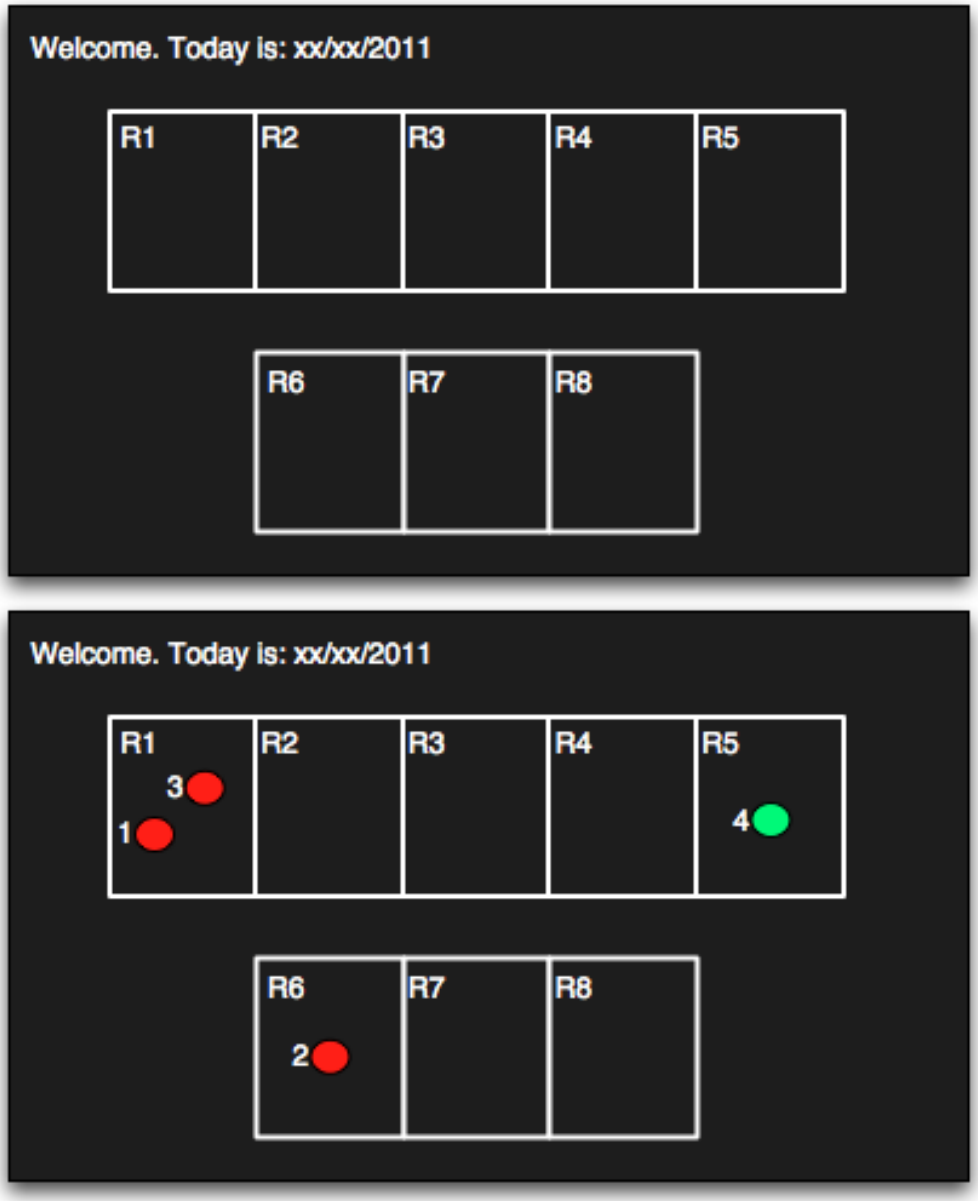
In the initial phase, activity records are printed on the console, we can only check after the system writes the log files. So in latter two prototype we design and implement user interfaces for the measurement system. The user interfaces for prototype one and prototype two don't sup-

port user interaction, then comes up the flexibility problems. So we design another user interface, this section describes both designs of the user interfaces.

User interface of prototype one & prototype two

The idea of this user interface is to show the trail of activities. We assume that activities are carried out in different rooms, and use one rectangle to represent a room that one Bluetooth device is placed inside it. When the system detects one allowed device, it will calculate the coordinate and paint a green dot and the step number. When the system leaves that room, it repaints the dot in red color, as shown in Figure 19, 'R' means "Room".

Figure 19 Prototype one & prototype two's user interface



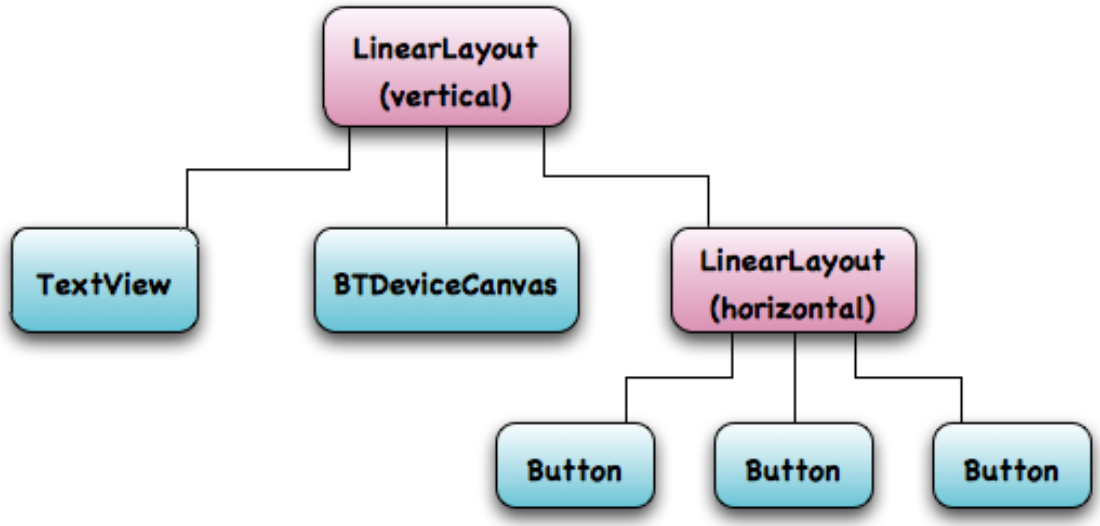
This user interface shows the location of the user clearly, thus, we can infer what activities have been done from those room numbers. In addition, it shows the steps of activities, we can read activity flow from the map directly. But, one disadvantage is that the map of rooms is predefined, which Bluetooth device should be put in which room is fixed, it cannot change the map of rooms according to real testing environment, and it doesn't support user interaction.

User interface of prototype three

Based on the testing and analysis of the user interface for prototype one and prototype two, we propose another user interface solution to represent the detecting results. This user interface supports user interactions that train the system to recognize allowed Bluetooth devices dynamically. We assume that one device represents a certain activity, so instead of showing rooms, this user interface paints devices.

The third prototype is developed for Android platform, so we take the advantage of declarative way to build user interface – using XML layout files, this separates user interface from business logic code completely, makes user interface replaceable. This design pattern is inspired by Web development, like HTML or XML files, which define the user interface layout in readable structure, this facilitates the development and debugging. Figure 20 shows the structure of prototype three's user interface layout.

Figure 20 Structure of user interface layout in prototype three

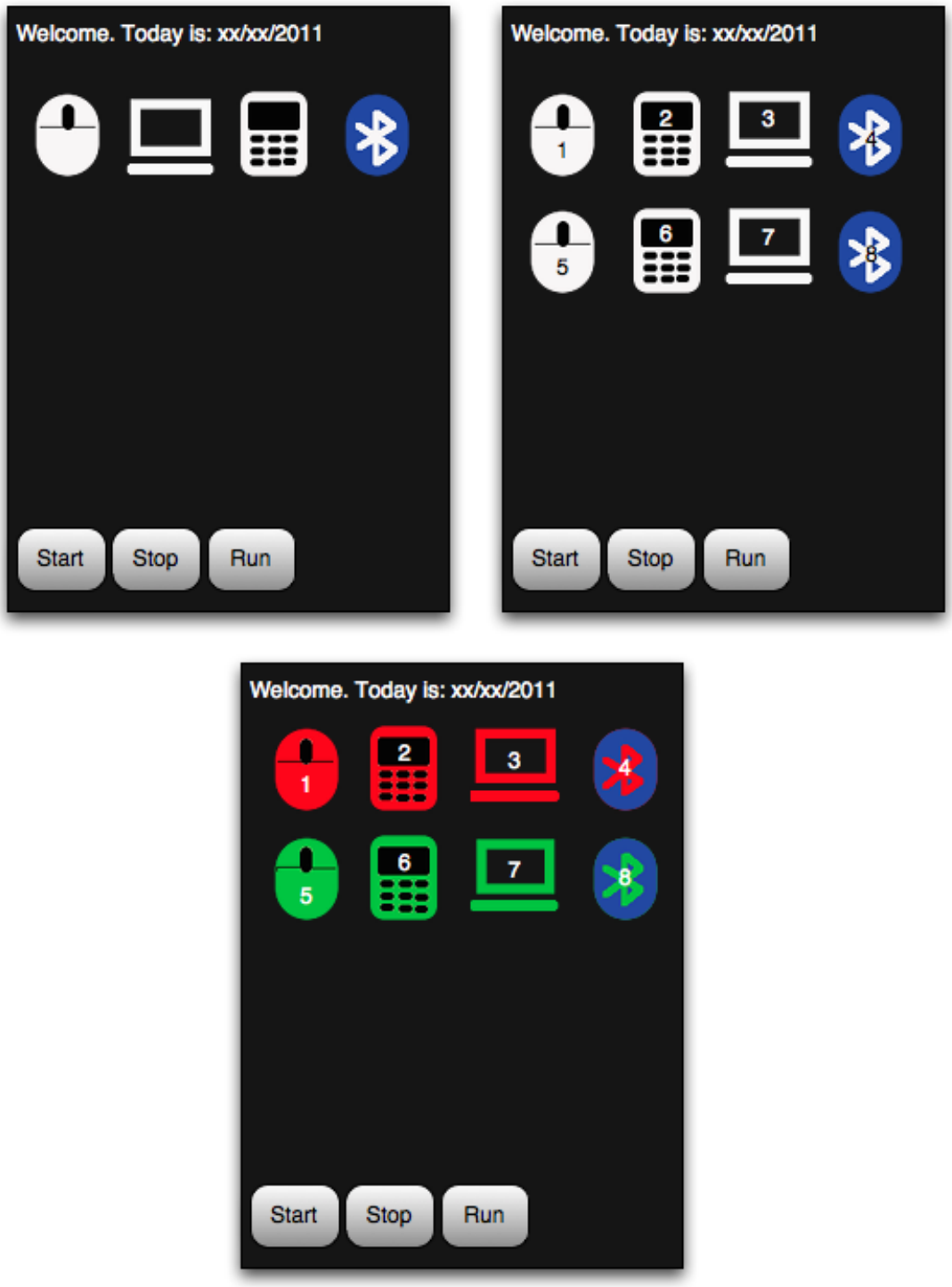


The user interface layout takes hierarchy structure, two classes are used in user interface layout, View and ViewGroup. View objects are widgets and ViewGroup objects are layouts, each XML element has its corresponding java object.

The structure of the layout file is simple, LinearLayout is a ViewGroup, TextView and Buttons are Android's predefined widgets, and BTDeviceCanvas is a self-defined ImageView. These elements will be initialized during the run-time when the system loads the user interface lay-

out. The main user interface is presented in Figure 21, a TextView on the top shows welcome information. On the bottom, button Start and button Stop are used for training allowed Bluetooth devices, and Run button will start recording activities. A self-defined ImageView in the middle shows the results of real-time data acquisition.

Figure 21 User interface of prototype three



ImageView is for displaying an arbitrary image, it can load images from various sources, can be used in any layout manager, and can easily scale and tint the image (Android Developer, 2011), so it is suitable for presenting the detecting result. When the system receives the response from remote Bluetooth device, it can get information of that Bluetooth device, such as its user-friendly name, Bluetooth address, and device class. To better represent the detecting results, different kinds of icons are designed to represent different types of Bluetooth devices, as shown in Figure 21, a mouse, a PC/laptop, a phone, or other Bluetooth device. Open state begins when the system is started. Then the system shows all in range Bluetooth devices (thesis will be considered as irrelevant devices for measurement). When user clicks Start button to trigger the system to change to training state, the system numbers all new devices that are detected during this state and stores their identifications as allowed Bluetooth devices. The allowed devices are used in measurement, but the devices detected before training state are considered as irrelevant and they are neither shown nor used in measurement. User clicks Run button to start measurement, which means recording event logs based on allowed devices. In this state, a green icon means that device is in system's detecting range, and in contrast, a red icon means the device is out of range. The enter and leave times of allowed devices are recorded in the event log.

5.5 Developed software

Three prototypes have been developed (see Table 5), prototype one is for mini-laptop runs on Ubuntu OS, we use Qt Creator and C++ programming language to develop it. Prototype two is for mobile phone runs on Maemo mobile OS, it is directly ported from prototype one by using Nokia Qt SDK. Prototype three is for Android mobile phone, we develop it in Java programming language by using Eclipse IDE with ADT plug-in.

Maemo is the simplified version of Debian GNU/Linux, so it is very easy to port the measurement system developed for Ubuntu Netbook Edition OS to it. In the study we use Nokia Qt SDK tool to customize our Linux program for Maemo mobile OS, especially optimize the user interface for mobile devices. Nokia Qt SDK builds on Qt Creator, its lightweight simulator brings great convenience for the development.

Android platform is also based on Linux kernel, but actually, it runs on Google's own Java virtual machine Dalvik. Thus, it is not possible to port Linux applications to Android platform directly. In this study, we use Eclipse with ADT plug-in to develop Android platform target-

Table 5 Comparison of developed prototypes

	Prototype 1	*Prototype 2*	*Prototype 3*
Running Device	Mini-laptop	Mobile Phone	Mobile Phone
Running Platform	Ubuntu Netbook Edition	Maemo 5	Android 2.1
Development Tool	Qt Creator	Qt Creator; Nokia Qt SDK	Eclipse; ADT plug-in
Programming Language	C++	C++	Java
Number of Modules	4	4	3
Software Size	≈ 700 lines	≈ 700 lines	≈ 1000 lines
Average Response Time (Enter)	6 seconds	7.64 seconds	2 seconds
Average Response Time (Leave)	17.3 seconds	20.7 seconds	18.86 seconds

ed system. The Android SDK also provides an emulator that simulates Android mobile devices, but there are several limitations of the emulator, the most critical one is that it doesn't support for Bluetooth. Thus, in the study, we have to use real Android phone to support the development by using USB debugging. Different from the emulator provided by Android SDK, the simulator in Nokia Qt SDK can well support Bluetooth.

Bluetooth APIs for Android platform are also different from the Bluetooth APIs that are used in Ubuntu and Maemo platforms. For Android platform, the first thing to user Bluetooth is to declare Bluetooth permission in the manifest file of the program. In Table 6, we compare Bluetooth APIs in different platforms. In Ubuntu/Maemo platforms, Host Controller Interface (HCI) provides a set of unified APIs to access low-level Bluetooth while Android provides several classes to management Bluetooth functionality. BluetoothAdapter class represents the local device Bluetooth adapter, BluetoothDevice class represents a remote Bluetooth device, BluetoothSocket class represents a connected or connecting Bluetooth socket, BluetoothClass. Device defines all device class constants and BluetoothClass.Device.Major defines all major device class constants (Android Developers, 2011).

Another difference is the results of Bluetooth discovery. In Ubuntu/Maemo platform, the hci_inquiry() function searches all in range Bluetooth devices, and returns the information of all those devices once. While in Android, discovery is an asynchronous call, it needs a BroadcastListener to listen to actions, such as 'ACTION_FOUND' and 'ACTION_DISCOVERY_FINISHED', so every time when one remote Bluetooth device is discovered, it will trigger the BroadcastListener, it gets discovery results one after another. For determine the enter time, it is more accurate in Android, when one device is found, it will trigger corresponding function immediately, don't need to wait for all the responses from in range devices.

Table 6 Comparison of Bluetooth APIs in three targeted platforms

	Ubuntu/Maemo	*Android*
Get local Bluetooth Dongle	Method: hci_get_route() hci_open_dev()	Class: BluetoothAdapter Method: getDefaultAdapter()
Get local Bluetooth Dongle Info	Method: hci_devinfo()	Class: BluetoothAdapter Method: getName() getState() getAddress()
Discovery remote Bluetooth devices	Method: hci_inquiry()	Class: BluetoothAdapter Method: startDiscovery() cancelDiscovery()
Discovery result	Address Info of all in range BT devices	Asynchronous call, BroadcastListener listen to DeviceFound action
Get remote BT device Info	hci_read_remote_name()	Class: BluetoothDevice Method: getAddress() getName() getBluetoothClass()
Recognize remote BT device type	None	Class: BluetoothClass.Device BluetoothClass.Device.Major

6 Testing

This chapter illustrates the laboratory testing and practical environment testing. Laboratory testing is conducted first to ensure the system do what we've designed correctly, and practical environment testing follows to validate whether the system can solve our research problem. For both testing, this chapter illustrates the testing environments, testing methods, testing procedures, and discusses the testing results, problems found during the testing, as well as solutions for these problems.

6.1 Laboratory environment testing

For laboratory testing, we conduct verification testing with white-box testing method. System's internal logic is tested to verify it meets the requirements that are determined in the use case diagram in Chapter 5: inquiring remote Bluetooth devices, acquiring Bluetooth device related information, determine the timestamps when does user enter or leave the radio proximity of a Bluetooth device, and recording activity logs into text files. Following is the detailed description of the laboratory testing.

Testing environment & configuration

Laboratory testing includes:

- Mini-laptop: Asus EEE PC 900. Memory: 1GB; Processor: 900MHz; Display: 8.9" Display, 1024 x 600 resolution.
- External Bluetooth adapter
- Bluetooth devices (e.g. Bluetooth mice, PC, phone, other types of BT devices)
- Operating system: Ubuntu Notebook Remix
- Integrated development environment software: Qt Creator

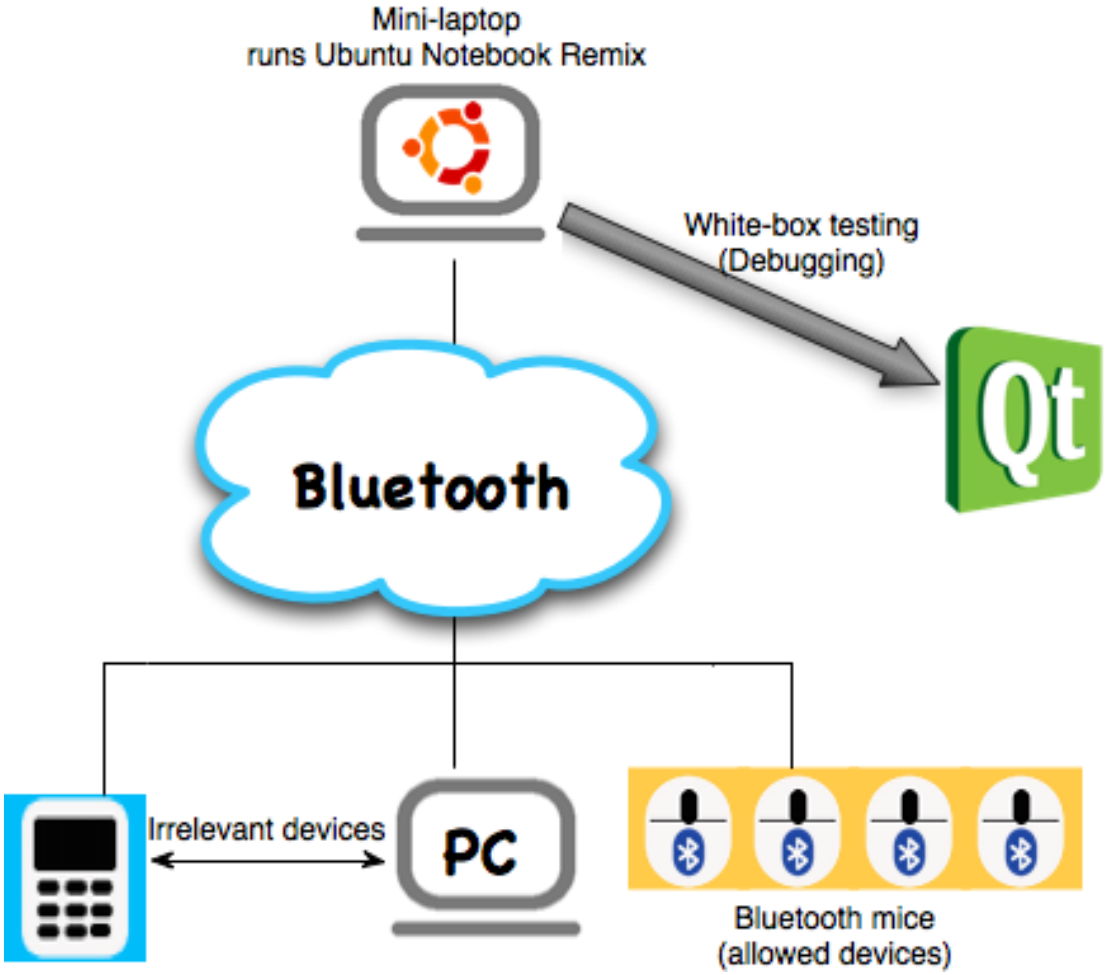
Figure 22 shows the structure of the laboratory testing environment. Asus EEE PC 900 mini-laptop is used for the testing, it runs Ubuntu Notebook Remix OS. An external Bluetooth adapter is needed to support the testing. Qt Creator is the IDE used to develop the system, and in the testing, it is used to conduct white-box testing – debugging and review the code. We use Bluetooth mice to simulate sensors, they act as allowed devices. Besides those, we also use other irrelevant Bluetooth devices to simulate a real testing environment.

Testing methods

Unit testing, which verifies the smallest unit of a software structure – module. It is a very important tool to verify the correctness of the code. Usually, it is carried out by programmers, so in this laboratory testing, we choose white-box testing method to review the code in detail. The advantage of white-box testing is that it tests the program thoroughly, and helps reveal bugs in the code.

There are two types of white-box testing. One is static analysis that test the system without run it. The other one is dynamic analysis, which runs the system with predefined inputs, and checks whether the system can output expected results. The dynamic analysis includes five

Figure 22 Laboratory testing setting



specific coverage testing methods: statement coverage, branch coverage, condition coverage, combination coverage, path testing.

Statement coverage requires enough test cases to ensure that each statement is executed at least once. Branch coverage requires test cases should cover all decision points, execute in both conditions: when the entire Boolean expression is true and when it is false. Condition coverage requires enough test cases to make every Boolean sub-expression in the decision point is executed both as true and false. Combination coverage requires all possibly Boolean combinations in the decision points should be executed at least once.

Testing procedure

First, determine functions for unit testing.

- Paint the map, as shown in Figure 19 in Chapter 5;
- Read allowed devices information from text file;
- Inquiry remote Bluetooth device, get Bluetooth address and other related information;

- Record the time when user enter the range of an allowed device, paint green dot to corresponding place on the map;
- Determine when does user leave the range of an allowed device, record the time, and replace the green dot with a red one;
- Write activity records into text files;
- Create different log file for different date.

Second, test cases are designed to meet the requirements of combination coverage testing approach (see Table 7 and Table 8).

Table 7 Test case one – recording enter time

Allowed	*Exist in record*	*Action*	*Path*
T	T	No action	isAllowed() -> isRecordExist()
T	F	Call addRecord()	isAllowed() -> isRecordExist() -> addRecord()
F	F	No action	isAllowed()

Table 8 Test case two – recording leave time

Is in scope	*Current discovered*	*Action*	*Path*
T	T	No action	inScope -> isDeviceExist()
T	F	Record leave time	inScope -> isDeviceExist() -> leaveRecord()
F	F	No action	inScope

Third, conduct the white-box test according to the designed test cases, identify and fix bugs. We simulate the testing environment in this way: the mini-laptop remains relatively stationary, we simulate its movement by switching on and off different Bluetooth devices. For example, we switch on No.1 Bluetooth mouse, which is predefined belong to Room 1, it simulates the scenario that the mini-laptop is entering Room 1 and detects No.1 Bluetooth mouse, and if we switch off NO.1 Bluetooth mouse, it simulates the scenario that the mini-laptop is leaving Room 1.

In the testing, we assume that we have 8 rooms, and the location of Bluetooth mice is predefined in the allowed devices text file. The only way to change it is to edit the text file manually, it is one of the limitations of prototype one. Figure 23 is the allowed devices list in laboratory testing, each line shows one Bluetooth device's device no., address, and the Room number. For instance, No.1 Bluetooth mouse, its address is 00:11:67:FA:1F:26 is in Room 1, and No.2 and No. 3 Bluetooth mice, their address are 00:11:67:FA:1F:36 and 00:11:67:FA:1E:2A both are in Room 2.

In real-life environment, the system will face many interferences from other irrelevant Bluetooth devices outside the system, so we also use other Bluetooth devices, such as mobile phones, PCs to simulate a real environment.

Testing results

The testing results are analyzed as follows:

- The system fulfills the basic functions requirements that are determined above, and some bugs are found and corrected.
- The system has a significant delay in detecting the enter time and leave time. The mean response time for detecting the enter movement is 6 seconds, and the mean response time for detecting the leave movement is 17.3 seconds. This influences the accuracy of activity's timestamps, so when we use those event logs to do automatic process modeling, we have to take the time delay into account.
- One problem in the testing is that those mice will be put into sleeping mode after about 3 minutes. When one mouse is brought into sleeping mode, it cannot be detected anymore, so the system will think this device is not in range. Later, we figure out a simple solution that is to use tape to keep the mouse active.
- The prototype cannot distinguish different types of Bluetooth device (e.g. a mouse, a phone, or PC).
- This laboratory testing is an ideal situation, we switch on and off Bluetooth mice, so there is no signal overlapping problems, but in real environment we have to face this problem. Two Bluetooth mice in two different rooms they might have signal overlap, and the system doesn't have solutions to discriminate the exact location of the user when there are many Bluetooth signals. We propose two possible ways, by comparing Link_Quality or Received Signal Strength Indication (RSSI), but both of them are Bluetooth vendor dependent, which means they are dependent on how Bluetooth module vendor implement the measurement of the link quality.

Figure 23 Predefined allowed device list

Device No.	Address	Room No.
1	00:11:67:FA:1F:26	1
2	00:11:67:FA:1F:36	2
3	00:11:67:FA:1E:2A	2
4	00:11:67:FA:1E:03	3
5	00:11:67:FA:20:3E	4
6	00:11:67:FA:1E:41	5
7	00:11:67:FA:20:1E	6
8	00:11:67:FA:1F:07	7
9	00:11:67:FA:1F:0B	7
10	00:11:67:FA:20:43	8

6.2 Practical environment testing

For prototype two, we choose Nokia N900 that runs Maemo OS as the testing device, and for prototype three, we use HTC Wildfire that runs Android OS. The purpose of practical environment testing is to validate whether the system can solve our research problem: can Bluetooth technology solution fulfill the requirements of acquisition real-time data for automatic process modeling.

Testing environment & configuration

Testing tools includes:

- Mobile phone for prototype two: Nokia N900. RAM: 256 MB; CPU: ARM Cortex A8 600 MHz; Display: 3.5 inches, 800 x 480 pixels.
- Mobile phone for prototype three: HTC Wildfire. RAM: 384 MB; CPU: ARM 11 processor 528 MHz; Display: 3.2 inches, 240 x 320 pixels.
- Bluetooth mice, and other Bluetooth devices from real environment outside the system.
- Operating system for prototype two: Maemo 5.
- Operating system for prototype three: Android OS, v2.1 (Eclair).

Figure 24 Practical environment testing setting

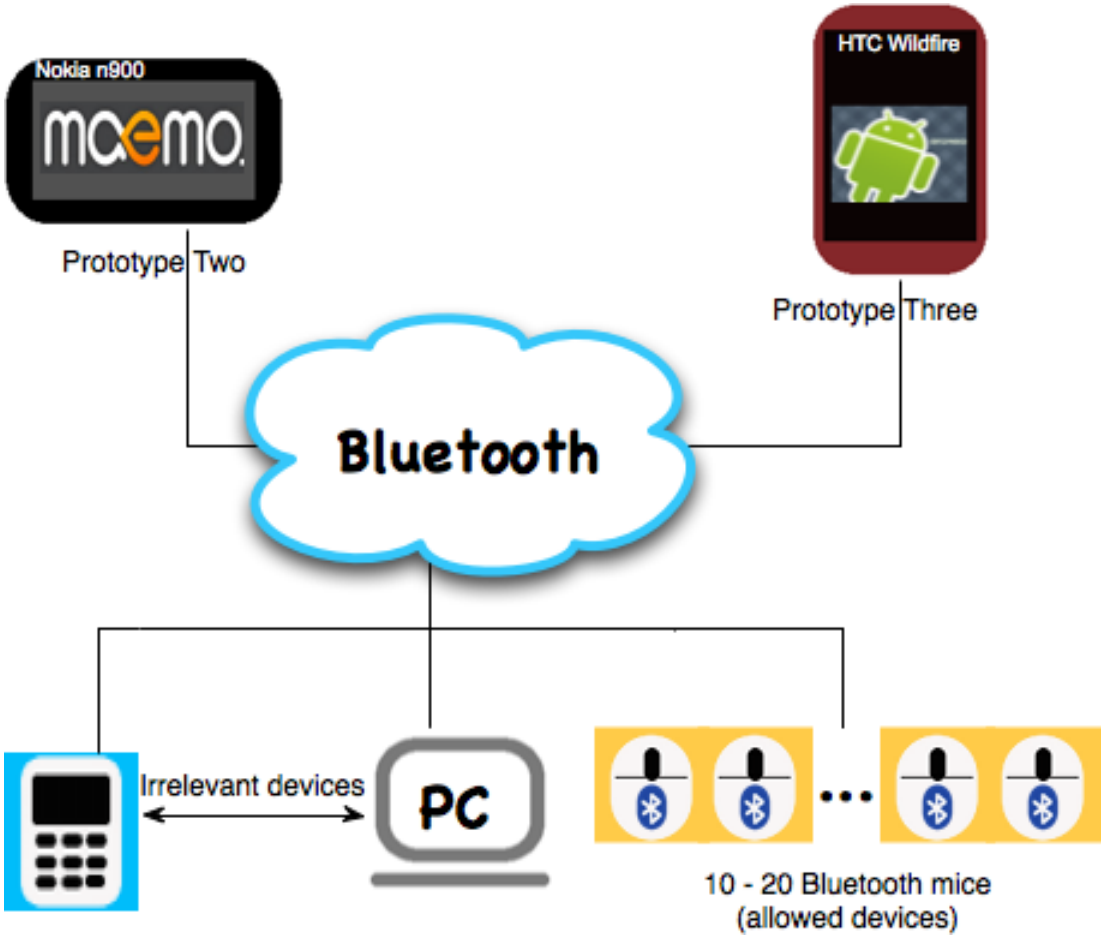
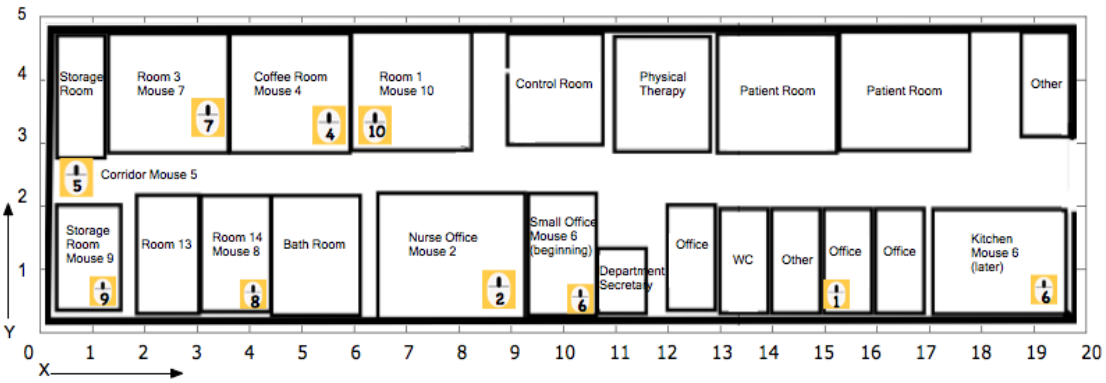


Figure 24 shows the structure of the practical environment testing. Prototype two is tested on Nokia N900, which runs Maemo 5 OS, and prototype three is tested on HTC Wildfire, which runs Android 2.1 OS. Both testing are conducted in Department 101 in the Helsinki University Central Hospital. Hospital District of Helsinki and Uusimaa (HUS) is Finland's largest university hospital district, and the Helsinki University Central Hospital is Finland's largest university hospital, it belongs to HUS (Collaborative Oncological Gene-environment Study, 2011).

Figure 25 is the map of Department 101 in the Helsinki University Central Hospital, it shows the allocation of those Bluetooth mice: Mouse 7 in Room 3, Mouse 4 in Coffee Room, Mouse 10 in Room 1, Mouse 9 in Storage Room, Mouse 8 in Room 14, Mouse 2 in Nurse Office, Mouse 6 in Small Office, Mouse 1 in the office at the right down corner of the map. We mapped Department 101 into a two-dimensional coordinate system for further analysis. As shown in Figure 25, there are x-axis and y-axis, both of them use centimeter as the measurement unit.

Figure 25 Map of department 101, Helsinki University Central Hospital



Testing schedule

Following is the schedule of practical environment testing:

- December, 2010, we test prototype two that runs on Nokia N900 in Department 101 in HUS, it is carried out by nurse in that department.
- Gather and analyze testing results, problems, and determine new requirements.
- Design and develop prototype three for Android platform.
- Later, prototype three, which runs on HTC Wildfire is also tested in Department 101 in HUS by nurse.

Testing results

About 200 activity records were collected from the testing in HUS in December, 2010. Figure 26 shows some of the records. The text log file records the Bluetooth address, the enter time and leave time, from the log file following problems are found and solved:

- Usually, in a practical environment it will create a large quantity of data in one day, but prototype two is designed to save whole day's records in one text file, it doesn't facilitate further analysis. This has been improved in prototype three, a folder will be created to save whole day's records, and the system will create a new log file every time when the application is started.
- In the log, it has only Bluetooth address, and the device number is predefined in allowed devices file. So we have to reference the allowed devices file frequently when analyzing the result, it increases the workload of analysis. This is also improved in prototype three, device information and timestamps information are stored in the same ArrayList, and are written into text file simultaneously.
- The allowed devices are predefined and written into a text file manually, it lacks flexibility. In this testing, the Bluetooth devices have to be set according to the predefined file. It is essential to make the system to be suitable for different running environments, so in prototype three, training allowed devices function is added.
- Different from the ideal environment in the laboratory testing, there are overlapping Bluetooth signals exist in the practical testing environment. From the log file, we found that the system can detect the Bluetooth mice that are set in other rooms, these redundant information influence the accuracy of event logs. Solutions are proposed and will be implemented in later prototype.

Figure 26 Result log file of practical environment testing in department 101


Bluetooth Address: 00:11:67:FA:20:1E; Enter Time: ti joulukuuta 21 13:54:06 2010; Leave Time: ti joulukuuta 21 13:55:07 2010;
Bluetooth Address: 00:11:67:FA:1E:2A; Enter Time: ti joulukuuta 21 13:54:06 2010; Leave Time: ti joulukuuta 21 13:55:07 2010;
Bluetooth Address: 00:11:67:FA:1E:03; Enter Time: ti joulukuuta 21 13:54:06 2010; Leave Time: ti joulukuuta 21 13:55:07 2010;
Bluetooth Address: 00:11:67:FA:1F:36; Enter Time: ti joulukuuta 21 13:54:06 2010; Leave Time: ti joulukuuta 21 13:55:07 2010;
Bluetooth Address: 00:11:67:FA:1F:07; Enter Time: ti joulukuuta 21 13:55:34 2010; Leave Time: ti joulukuuta 21 13:56:11 2010;
Bluetooth Address: 00:11:67:FA:1F:0B; Enter Time: ti joulukuuta 21 13:55:34 2010; Leave Time: ti joulukuuta 21 13:56:11 2010;
Bluetooth Address: 00:11:67:FA:20:43; Enter Time: ti joulukuuta 21 14:00:35 2010; Leave Time: ti joulukuuta 21 14:01:33 2010;
Bluetooth Address: 00:11:67:FA:1E:41; Enter Time: ti joulukuuta 21 14:00:35 2010; Leave Time: ti joulukuuta 21 14:01:33 2010;
Bluetooth Address: 00:11:67:FA:1E:03; Enter Time: ti joulukuuta 21 14:05:23 2010; Leave Time: ti joulukuuta 21 14:05:53 2010;
Bluetooth Address: 00:11:67:FA:1E:03; Enter Time: ti joulukuuta 21 14:06:23 2010; Leave Time: ti joulukuuta 21 14:06:36 2010;
Bluetooth Address: 00:11:67:FA:1E:03; Enter Time: ti joulukuuta 21 14:08:10 2010; Leave Time: ti joulukuuta 21 14:08:41 2010;
Bluetooth Address: 00:11:67:FA:1E:03; Enter Time: ti joulukuuta 21 14:08:59 2010; Leave Time: ti joulukuuta 21 14:09:12 2010;
Bluetooth Address: 00:11:67:FA:1E:03; Enter Time: ti joulukuuta 21 14:09:29 2010; Leave Time: ti joulukuuta 21 14:09:41 2010;
Bluetooth Address: 00:11:67:FA:1E:03; Enter Time: ti joulukuuta 21 14:10:00 2010; Leave Time: ti joulukuuta 21 14:10:13 2010;

Prototype three improves the prototype two, a preliminary test is conducted in ETLA, the Research Institute of the Finnish Economy before testing the system in Department 101 in HUS, and several problems of prototype three were found:

- The screen of Android phone will timeout after one minute, and then arouse the stall problem, solution could be: set screen lock never go to sleep.
- Another problem is that with touch screen, it is easy to click other buttons (such as menu or back button) by accident, the application usually stall when it is active again. The reason is that Bluetooth driver is busy when you accidentally stop the application and then restart it. So the solution could be reboot phone's Bluetooth driver when the application is started.

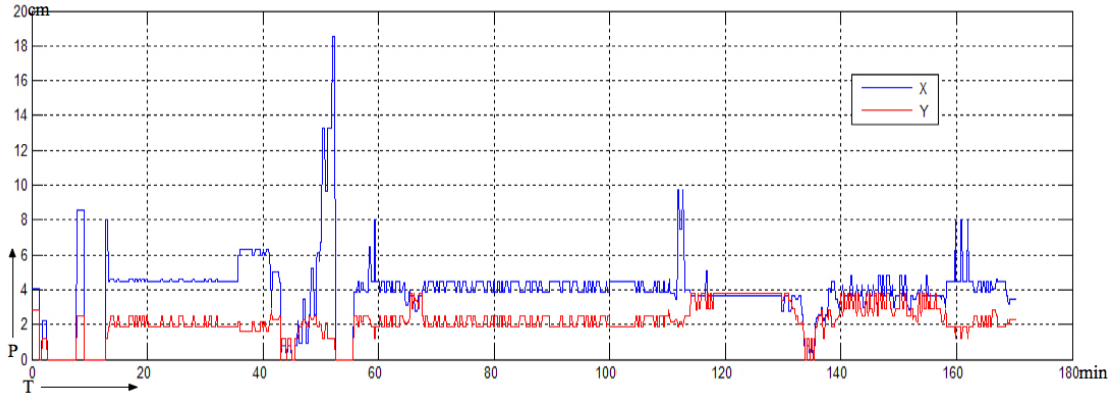
Result analysis

Then we plotted those records in the log file to make a line graph, T-axis means timeline, use minutes as measurement unit, and P-axis shows the position (the value on x-axis and value on y-axis in Figure 25). For every record, we will calculate the activity's position in the coordinate system in Figure 25, we use red points to represent the value on x-axis, blue points to represent the value on y-axis. The result of the analysis is shown in Figure 27, it shows the movements of a nurse.

To ensure the accuracy of process information extracted from log files, the measurement system needs more improvements. We should measure Bluetooth link quality to approximate the distance between the device that runs the measurement system and other remote Bluetooth devices. Then, we can solve Bluetooth device signal overlapping problems, this could be implemented as follows:

As mentioned in Chapter 5, we used BroadcastListener class to listen to two actions, 'ACTION_FOUND' and 'ACTION_DISCOVERY_FINISHED' when the system is inquiring. The former action means a remote Bluetooth device is discovered, it includes several extra fields, such as EXTRA_DEVICE that represents a BluetoothDevice class instance, EXTRA_CLASS represents a BluetoothClass class instance, EXTRA_NAME is the user-friendly name of the

Figure 27 Calculation of the movement of nurse (calculated by Valeriy Naumov)



remote Bluetooth device. And we can use EXTRA_RSSI filed, it tells the RSSI value of the remote Bluetooth device. RSSI is a signed 8-bit value, it is reported by Bluetooth hardware, and can indicate the strength of arriving signal.

Then we will compare the RSSI values of all in range Bluetooth devices, to estimate which Bluetooth device is the measurement system is closer to. This would improve the accuracy of the real-time data collected by the system greatly.

7 Conclusions and plans of further research

Population aging is global and unalterable trend, the VESC project in Finland is found to develop smart living environment for senior citizen. Its objective is to help the elderly people live more independent lives and improve the quality of elderly healthcare, so there are needs to study the behaviors of the elderly as well as the behaviors of nurses in open care processes. This research has been conducted to develop and test a prototype for the need of collecting real-time data for automatic process measurement.

In this research, we solved both of our research questions. We successfully applied Bluetooth communication technology in the developed prototypes, which collect real-time process data for automatic process measurement. We have combined together the concepts of smart living environment and process management. We have proposed the automatic process measurement method based on the idea of process mining. We used software prototyping combined with iterative software development process approaches in our research, and three iterations were conducted. The results of three iterations are three prototypes run on different platforms, in the first iteration, we built prototype one for mini-laptop that runs Ubuntu OS, then we customized the prototype for mobile phone and port it to Maemo mobile OS. Besides, we developed another prototype for Android platform. We have tested the system, in both laboratory environment and practical environment. Through developed prototypes, we studied different Bluetooth APIs for different Linux platforms. We compared different development tools as well as targeted platforms, this will contribute to our further researches.

Several problems were found during the practical environment testing in Department 101 in HUS, and these problems were solved in the development of prototype three for Android platform, we improved the flexibility of the system by adding training allowed Bluetooth devices function. Some problems of prototype three were also found in the preliminary testing conducted in ETLA, such as stall problems, we have figured out solutions and the system can be improved in next iteration of development.

According to the analysis of the testing results, the developed measurement system achieves collecting the basic data needed for automatic process measurement. However, new problems and requirements were found during the experiment, and further research is still needed after this work:

Bluetooth link quality. In further research, we should ensure the measurement system collect more accurate real-time data, so it is critical to solve Bluetooth signal overlapping problems. The most feasible solution could be using Bluetooth link quality value to approximate the distance between the device running the measurement system and other remote Bluetooth devic-

es. The Bluetooth APIs for Android platform provides the RSSI value of the remote Bluetooth device, this value indicates the strength of the arriving signal. Then, we can estimate the location and reduce those redundant records, thus, improve the accuracy of real-time data collected by the measurement system.

Support multiple sites measurement and multiple measurement devices. In reality, there are many sites need to measure, so it requires different sets of allowed devices for different measurement environments. This means the system should be possible to train several sets of allowed devices and save their information in different text files. Users can choose which set of allowed devices they are going to use in the measurement, as shown in Figure 28.

In current experiments, only one measurement device was used, but in further research, many measurement devices could be used in testing. So the system should support sharing allowed devices lists between different measurement devices, as shown in Figure 29. Another problem would be the automatic process modeling, by which process models are extracted from these event logs, but this is out of the scope of the present work.

Figure 28 Solution for multiple site measurements

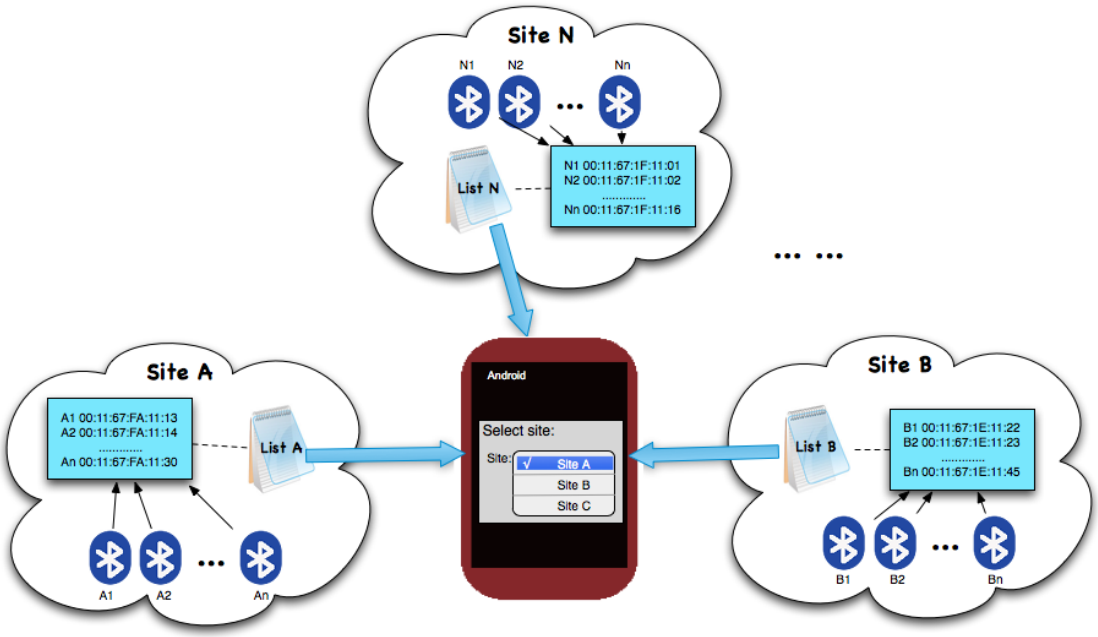
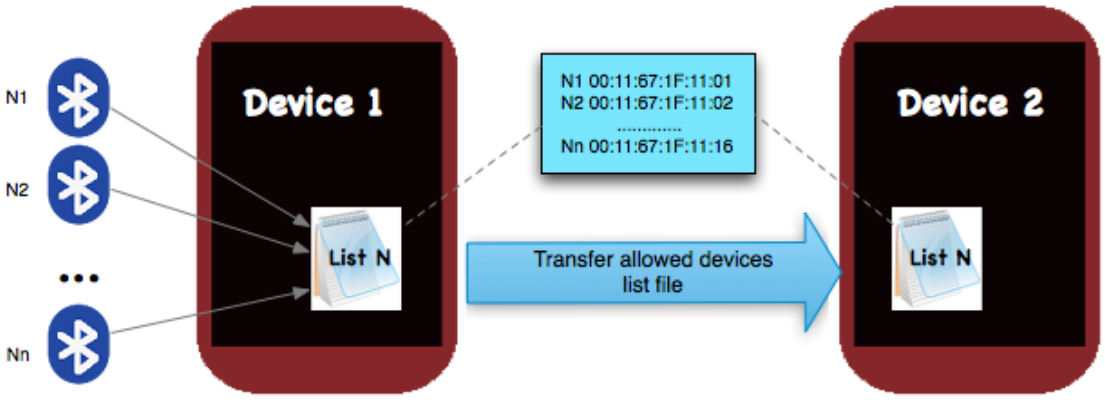


Figure 29 Solution for sharing allowed devices



References

Academy of Finland website. (visited 17.2.2011). Retrieved from: http://www.aka.fi/Tiedostot/Tiedostot/MOTIVE/Hankekuvaukset/vesc-conso.pdf

Affix web site. (visited 1.2.2011). Retrieved from: http://affix.sourceforge.net

Agrawal, R., Gunopuloa, D., & Leymann, F. (1998). *Mining Process Models from Workflow Logs.* Proceeding of the 6th International Conference on Extending Database Technology, 469–483.

Android Developers website. (visited 24.2.2011). Retrieved from: http://developer.android.com/index.html

Bhagwat, P. (2001). *Bluetooth: Technology for Short-range Wireless Apps.* Internet Computing, IEEE, vol.5, no.3, pp. 96–103. doi: 10.1109/4236.935183

Bisdikian, C. (2001). *IBM Research Report: An Overview of the Bluetooth Wireless Technology.* Communications Magazine, IEEE, vol. 39, no. 12, pp. 86–94. doi: 10.1109/35.968817

Bluetooth Org. (2010). Bluetooth Specification Version 4.0, vol.0. Retrieved from: http://www.Bluetooth.com

BlueZ Org website. (visited 1.2.2011). Retrieved from: http://www.bluez.org

Chan, M., Hariton, C., Ringeard, P., & Campo, E. (1995). *Smart house automation system for the elderly and the disabled.* Systems, Man and Cybernetics, Intelligent Systems for the 21st Century. IEEE International Conference, vol. 2, pp. 1586–1589. doi: 10.1109/ICSMC.1995.537998

Changrui Ren, Wei Wang, Jin Dong, Hongwei Ding, Bing Shao, & Qinhua Wang. (2008). *Towards a flexible business process modeling and simulation environment.* Simulation Conference, WSC, pp. 1694–1701. doi: 10.1109/WSC.2008.4736255

Collaborative Oncological Gene-environment Study website. (visited 17.2.2011). *Participant 8 – The Hospital District of Helsinki and Uusimaa (HUS)*. Retrieved from: http://www.cogseu.org/

Cook, J.E. & Wolf, A.L. (1995). *Automating Process Discovery through Event-data Analysis.* Seattle, Washington, USA, Proceeding of the 17th International Conference on Software Engineering, 73–82.

Davenport, T. (1993). Process Innovation: Reengineering work through information technology. Boston: Harvard Business School Press.

Ferro, E. & Fotorti, F. (2005). Bluetooth and Wi-Fi wireless protocols: A survey and a comparison. Wireless Communications, IEEE, vol. 12, no.1, pp. 12–16. doi: 10.1109/MWC.2005.1404569

Hori, T., Nishida, Y., Aizawa, H., Murakami, S., & Mizoguchi, H. (2004). *Sensor Network for Supporting Elderly Care Home.* Sensors, Proceedings of IEEE, vol. 2, 24–27, pp. 575–578. doi: 10.1109/ICSENS.2004.1426230

Hwang, J.Y., Kang, J.M., Jang, Y.W., & Kim, H.C. (2004). *Development of novel algorithm and real-time monitoring ambulatory system using Bluetooth module for fall detection in the elderly.* Engineering in Medicine and Biology Society, IEMBS '04. 26th Annual International Conference of the IEEE, vol. 1, pp. 2204–2207. doi: 10.1109/IEMBS.2004.1403643

Jin-Shyan Lee, Yu-Wei Su, & Chung-Chou Shen. (2007). *A Comparative Study of Wireless Protocols: Bluetooth, UWBM, ZigBee, and Wi-Fi.* Industrial Electronics Society. IECON, 33rd Annual Conference of the IEEE, pp. 46–51. doi: 10.1109/IECON.2007.4460126

Jovanov, E., O'Donnell Lords, A., Raskovic, D., Cox, P.G., Adhami, R., & Andrasik, F. (2003). *Stress monitoring using a distributed wireless intelligent sensor system.* Engineering in Medicine and Biology Magazine, IEEE, vol. 22, no.3, pp. 49–55. doi: 10.1109/MEMB.2003.1213626

Kansal, A. (2002). *Bluetooth Primer.* Los Angeles: Red-M.

Kardach, J. (2000). *Bluetooth Architecture Overview.* Intel Technology Journal, (Q2):7. Retrieved from: http://www.kardach.com/Bluetooth/runic_stone_journal/Entries/2007/9/2_Bluetooth_Architecture_Overview_files/Bluetooth%20Arch%20ppt.pdf

Li Yan & Feng Yu-qiang. (2006). *An Automatic Business Process Modeling Method Based on Markov Transition Matrix in BPM.* Management Science and Engineering, ICMSE 'O6, 2006 International Conference, pp. 46–51. doi: 10.1109/ICMSE.2006.313842

Maemo Org website. *Bluetooth.* (2010). Retrieved from: http://wiki.maemo.org/Bluetooth

Maemo Org website. *Software Platform.* (visited 10.2.2011). Retrieved from: http://maemo.org/intro/platform/

Maemo Org website. *Qt4 Development. (2010). Retrieved from:* http://wiki.maemo.org/Qt4_Development#Porting_Qt_applications_to_Maemo

Mainwaring, A., Polastre, J., Szewezyk, R., Culler, D., & Anderson, J. (2002). *Wireless Sensor Network for Habitat Monitoring.* New York: the 1st ACM International Workshop on Wireless Sensor Networks and Applications, pp. 88–97. doi: 10.1145/570738.570751

Miller, M. (2001). *Discovering Bluetooth*. Sybex Inc. ISBN-10: 0782129722

Noury, N. (2002). *A smart sensor for the remote follow up of activity and fall detection of the elderly.* Microtechnologies in Medicine & Biology 2nd Annual International IEEE-EMB Special Topic Conference, pp. 314–317. doi: 10.1109/MMB.2002.1002337

Park, C., Liu, J., Chou, P.H. (2005). *Eco: an ultra-compact low-power wireless sensor node for real-time motion monitoring.* Information Processing in Sensor Networkds. IPSN, 4th International Symposium, pp. 398–403. doi: 10.1109/ISPN.2005.1440956

Pollack, M.E., Brown, L., Colbry, D., Orosz, C., Peintner, B., Ramakrishnan, S., Engberg, S., Matthews, J. T., Dubar-Jacob, J., McCarthy, C.E., Thrun, S., Montemerlo, M., Pineau, J., & Roy, N. (2002). *Pearl: A mobile robotic assistance for the elderly.* AAAI Technical Report WS-02-02.

Population Division, Department of Economic and Social Affairs, The United Nations. (2010). *World Population Aging 2009.*New York: United Nations Publication.

Qt website. (visited 10.2.2011). Retrieved from: http://qt.nokia.com/

Rathi, S. (2000). *Blue Tooth Protocol Architecture.* Dedicated Systems Magazine – 2000 Q4. Retrieved from: http://www.es2.be/magazine/00q4/2000q4_p028.pdf

Sairam, K.V.S.S.S.S., Gunasekaran, & N., Reddy, S.R. (2002). *Bluetooth in Wireless Communication.* Communications Magazine, IEEE, vol. 40, no. 6, pp. 90–96. doi: 10.1109/MCOM.2002.1007414

Santos, v., Bartolomeu, p., Fonseca, J., & Mota, A. (2007). *B-Live A Home Automation System for Disabled and Elderly People.* Industrial Embedded Systems, International Symposium, pp. 333–336. doi: 10.1109/SISE.2007.4297355

Smith, A. (1776). *The Wealth of Nations*. Cannan ed. New York: Modern Library.

Stallings, W. (2005). *Wireless Communications & Networks.* Pearson Prentice Hall. SBN-10: 0131918354, SBN-13: 9780131918351.

Tian He, Stankovic, J.A., Chen-Yang Lu, & Abdelzaher, T. (2003). *SPEED: a stateless protocol for real-time communication in sensor networks.* Distributed Computing Systems, 23rd International Conference, pp. 46–55. doi: 10.1109/ICDCS.2003.1203451

Ubuntu Documentation. *Installation System Requirements*. (visited 10.2.2011). Retrieved from: https://help.ubuntu.com/community/Installation/SystemRequirements

Ubuntu website. (visited 10.2.2011). Retrieved from: http://www.ubuntu.com/netbook

Van Der Aalst, W.M.P., Schonenberg, M.H., & Song, M. (2011). *Time prediction based on process mining.* Information Systems, Scopus, vol. 36, no. 2, pp. 450–475.

Wikipedia. *Bluetooth Special Interest Group*. (visited 10.2.2011). Retrieved from: http://en.wikipedia.org/wiki/Bluetooth_Special_ Interest_Group

Wikipedia. *Bluetooth.* (visited 17.2.2011). Retrieved from: http://en.wikipedia.org/wiki/Bluetooth

Xu, N., Rangwala, S., Chintalapudi, K., Ganesan, D., Broad, A., Govindan, R., & Estrin, D. (2004). *A Wireless Sensor Network For Structural Monitoring.* Baltimore, USA: Proc. ACM SenSys '04.

Aikaisemmin ilmestynyt ETLAn Keskusteluaiheita-sarjassa

Previously published in the ETLA Discussion Papers Series

No 1235 *Heli Koski – Mika Pajarinen*, Do Business Subsidies Facilitate Employment Growth? 04.01.2011. 20 p.

No 1236 *Antti-Jussi Tahvanainen – Raine Hermans*, Making Sense of the TTO Production Function: University Technology Transfer Offices as Process Catalysts, Knowledge Converters and Impact Amplifiers. 11.01.2011. 40 p.

No 1237 *Jukka Lassila – Tarmo Valkonen*, Julkisen talouden rahoituksellinen kestävyys Suomessa. 11.01.2011. 28 s.

No 1238 *Martin Kenney – Bryan Pon*, Structuring the Smartphone Industry: Is the Mobile Internet OS Platform the Key. 10.02.2011. 24 p.

No 1239 *Mika Maliranta – Reijo Mankinen – Paavo Suni – Pekka Ylä-Anttila*, Suhdanne- ja rakennekriisi yhtä aikaa? Toimiala- ja yritysrakenteen muutokset taantumassa. 17.02.2011. 20 s.

No 1240 *Jyrki Ali-Yrkkö – Petri Rouvinen – Timo Seppälä – Pekka Ylä-Anttila*, Who Captures Value in Global Supply Chains? Case Nokia N95 Smartphone. 28.02.2011. 22 p.

No 1241 *Antti Kauhanen – Sami Napari*, Gender Differences in Careers. 9.03.2011. 31 p.

No 1242 *Mika Pajarinen – Petri Rouvinen – Pekka Ylä-Anttila*, Omistajuuden vaikutus suomalaisen työllisyyden kasvuun ja pysyvyyteen. 16.03.2011. 27 s.

No 1243 *Rita Asplund – Sami Napari*, Intangibles and the Gender Wage Gap. An Analysis of Gender Wage Gaps Across Occupations in the Finnish Private Sector. 22.03.2011. 2 p.

No 1244 *Antti Kauhanen – Sami Napari*, Career and Wage Dynamics. Evidence from Linked Employer-Employee Data. 25.03.2011. 28 p.

No 1245 *Kari E.O. Alho*, Should Sweden Join the EMU? An Analysis of General Equilibrium Effects through Trade. 06.04.2011. 16 p.

No 1246 *Heli Koski – Mika Pajarinen*, The Role of Business Subsidies in Job Creation of Start-ups, Gazelles and Incumbents. 07.04.2011. 21 p.

No 1247 *Antti Kauhanen*, The Perils of Altering Incentive Plans. A Case Study. 08.04.2011. 22 p.

No 1248 *Rita Asplund – Sami Napari*, Intangible Capital and Wages. An Analysis of Wage Gaps Across Occupations and Genders in Czech Republic, Finland and Norway. 11.04.2011. 18 p.

No 1249 *Mari Kangasniemi – Antti Kauhanen*, Performance-related Pay and Gender Wage Differences. 21.04.2011. 19 p.

Elinkeinoelämän Tutkimuslaitoksen julkaisemat "Keskusteluaiheita" ovat raportteja alustavista tutkimustuloksista ja väliraportteja tekeillä olevista tutkimuksista. Tässä sarjassa julkaistuja monisteita on mahdollista ostaa Taloustieto Oy:stä kopiointi- ja toimituskuluja vastaavaan hintaan.

Papers in this series are reports on preliminary research results and on studies in progress. They are sold by Taloustieto Oy for a nominal fee covering copying and postage costs.

Julkaisut ovat ladattavissa pdf-muodossa osoitteessa: www.etla.fi/julkaisuhaku.php
Publications in pdf can be downloaded at www.etla.fi/eng/julkaisuhaku.php

ETLA
Elinkeinoelämän Tutkimuslaitos
The Research Institute of the Finnish Economy
Lönnrotinkatu 4 B
00120 Helsinki

Puh. 09-609 900
Fax 09-601 753
www.etla.fi
etunimi.sukunimi@etla.fi

ISSN 0781–6847