

# **Faculty of Electrical Engineering**

# FRAMEWORK OF DISTRIBUTED CONTROLLING VIA SMARTPHONE FOR INDUSTRIAL AUTOMATION APPLICATION

Phua Ging Sheng

Master of Science in Mechatronic Engineering

2013

# FRAMEWORK OF DISTRIBUTED CONTROLLING VIA SMARTPHONE

## FOR INDUSTRIAL AUTOMATION APPLICATION

PHUA GING SHENG

A thesis submitted in fulfillment of the requirements for the degree of Master of Science in Mechatronic Engineering

**Faculty of Electrical Engineering** 

## UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2013

n in the second second

## DECLARATION

I declare that this thesis entitle "Framework of Distributed Controlling via Smartphone for Industrial Automation Application" is the result of my own research except as cited in the references. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature	:	
Name	:	PHUA GING SHENG
Date	:	3 mac 2014



C Universiti Teknikal Malaysia Melaka

## ABSTRACT

Programming language that is used in industrial is platform specific, when hardware is updated or replaced by a newer hardware, the entire binary software needed to be changed and modified. Software that has been written is not able to run on different hardware platform without recompiling the source code. Besides that, the same code that is executed on machine is not able to be executed on computer for any hardware simulation or testing. Machine running code is not compatible with computer running code, thus, different types of source code are needed for computer simulation and real time platform respectively. This research was focused on extending Smartphone and Java platform into automation environment where users are able to monitor and control the targeted system regardless of any location by utilizing internet. The main task of this research had been working on creating new framework of distributed controlling and computing via smartphone. Platform independent frameworks of classes that implement internet protocol were designed to provide interaction over client server paradigm and finally all frameworks were then integrated over a system which allows access to automation control, manipulate and monitor via smartphone over wireless internet network protocol. Through this approach, Smartphone and Java platform's extendibility and adaption in different environment with Java features inherited were proven successfully. As a result, Industrial automation can use Java to produce systems that perform more reliably, improve programmer productivity, time efficiency and even provide binary compatibility across hardware platforms. Using Java, when comes to hardware simulation, the same application class files that run on the target machine can also execute on a desktop computer for development and debugging due to its platform independent ability. It simplifies remote monitoring and controlling in several ways where its internetworking based language allows it to run on any web browser or any other networking platform. In the event of a hardware fault, a developer could also write a dynamic Java classes that monitor a set of sensors on the hardware for maintenance and troubleshooting. Not only can dynamic troubleshoot the behavior of the hardware, but it can also be used to patch existing code or add additional functions without stoppage or modification of machine. The ability to change a single source file, recompile it and then dynamically download only the changed class file and link it with a running JVM provides added flexibility for software maintenance. Hardware independence ability allows developers to simulate the run-time environment on a desktop computer, greatly reducing development time and debugging is required only on a single version of source code, once for all platforms. This reduces the project overhead and maintenance by greatly simplifying project build configurations, reducing coding errors and simplifying bug tracking and change requests.

#### ABSTRAK

Bahasa pengaturcaraan yang digunakan dalam industri adalah platform tertentu, apabila perkakasan dikemaskini atau digantikan dengan perkakasan baru, perisian keseluruhan perlu diubah suai. Perisian yang telah ditulis tidak dapat berfungsi di platform yang berbeza tanpa dibina semula. Selain itu, kod yang sama yang dilaksanakan pada mesin tidak dapat dilaksanakan pada komputer untuk mana-mana perkakasan atau simulasi ujian. Kod mesin tidak serasi dengan kod komputer, oleh itu, pelbagai jenis kod sumber diperlukan untuk simulasi komputer dan platform sebenar.Kajian ini adalah untuk melanjutkan telefon pintar dan platform Java di dalam automasi dan pengguna boleh memantau dan mengawal apa-apa di mana-mana dengan hanya menggunakan internet. Tumpuan utama kajian ini adalah mereka rangka kerja baru untuk pengawal teragih dan pengkomputeran melalui telefon pintar. Kelas-kelas rangka kerja platform berdikari yang berasaskan protokol internet telah direka untuk menyediakan interaksi paradigma klien pelanggan. Semua rangka kerja dan perisian telah diintegrasikan ke atas sistem yang membolehkan akses kepada kawalan, pantauan, dan manipulasi automasi menggunakan telefon pintar melalui rangkaian protokol internet tanpa wayar. Melalui pendekatan ini, peranti tertanam isirong dan kesesuaian platform Java dan penggunaannya dalam persekitaran dan bidang yang berbeza dengan ciri-ciri Java diwarisi telah berjaya dibuktikan. Hasilnya, automasi perindustrian boleh menggunakan Java untuk menghasilkan sistem yang lebih tepat, meningkatkan produktiviti pengaturcara, kecekapan masa dan juga menyediakan keserasian binari di seluruh platform perkakasan. Kod aplikasi yang berjalan pada mesin sasaran juga boleh digunakan pada komputer untuk pembangunan dan debugging kerana kemampuannya untuk berfungsi di platform yang berbeza. Sekiranya suatu kesalahan perkakasan, pemaju juga boleh menulis dinamik kelas Java yang memantau satu set sensor pada perkakasan bagi penyelenggaraan dan penyelesaian masalah. Ia juga boleh digunakan untuk menampal kod yang sedia ada atau menambah fungsi-fungsi tambahan tanpa pemberhentian atau pengubahsuaian mesin. Keupayaan untuk menukar fail sumber tunggal, susun semula dan kemudian dinamik turun hanya fail kelas berubah dan menghubungkannya dengan JVM berjalan menyediakan fleksibiliti tambahan untuk penyelenggaraan perisian. Keupayaan platform bebas membolehkan pencipta mengurangkan penyelenggaraan, memudahkan projek konfigurasi, mengurangkan kod kesilapan dan memudahkan pengesanan pepijat dan perubahan permintaan.

v

## ACKNOWLEDGEMENT

In preparing this thesis, I was in contact with many people, researchers, academicians and practitioners. They have contributed towards my understanding and thought. I would like to take this opportunity to express my greatest appreciation to those people who have given their help, supports, and guidance during research.

In particular, I wish to express my sincere appreciation to my supervisors, Dr. Tay Choo Chuan and Associate Prof. Dr. Zulkifilie Bin Ibrahim for advices, critics, and guidance. Without their continued support and interest, this thesis would not have been completed.

Besides, I would like to thanks to the authority of UTeM for providing me with a good environment and facilities. My sincere appreciation also extends to all my colleagues and others who have provide assistance at various occasions. Their view and tips are useful indeed. Unfortunately, it is not possible to list all of them in this limited space. Thanks again.

# LIST OF TABLES

TABLE

## TITLE

## PAGE

2.1	Java main component	27
2.2	Function of Distributed Object Applications	32
2.3	Java networking processor features	38
3.1	Front component key point	52
3.2	Function and States of Front Component	54
3.3	Method Summary API	63
3.4	Method Summary API	75
3.5	Method Summary API	92
4.1	Debugging aid	107
4.2	Performance tuning	110
4.3	Design principle	112
5.1	Motor speed and current measurement A	126
5.2	Motor speed and current measurement B	127

## LIST OF FIGURES

FIGURE

## TITLE

## PAGE

1.1	Simplified single system overview	5
2.1	Trends changing	21
2.2	Development process	25
2.3	Platform independent	26
2.4	Platform architecture	28
2.5	RMI interaction	33
2.6	Mapping Pattern	43
2.7	Hibernate Architecture	44
3.1	Front Component Architecture	48
3.2	Last in first out stack	49
3.3	Front component lifecycle	51
3.4	Lifecycle callback	53
3.5	Resumed state	56
3.6	Paused state	57
3.7	Stopped state	58
3.8	Last in First Out Flow	59

3.9	Backstack flow	60
3.10	Multitasking	61
3.11	Client Component Architecture	65
3.12	Socket connection	67
3.13	Connection paradigm	69
3.14	Message manipulation method	70
3.15	Entity manipulation method	71
3.16	Server Component Architecture	81
3.17	Sending request	85
3.18	Creating object	85
3.19	Creating threads	86
3.20	Calling method	86
3.21	Sending response	87
3.22	Garbage collecting	87
3.23	Server component life cycle	89
4.1	Client side	100
4.2	Server side	101
4.3	Server side and automation	102
4.4	Complete system architecture	104
4.5	Multi request	105
5.1	Communication flow	117
5.2	Overall simplified demonstration diagram	120
5.3	Isolating circuit	122
5.4	Darlington transistor to drive circuit	124
5.5	Before and after - motor	125

5.6	Tachometer	125
5.7	Measurement 1 - Voltage vs speed	126
5.8	Measurement 2 - Voltage vs speed	127
5.9	Switching lighting system	129
5.10	Before and after - Lighting	132
5.11	Before and after - Sound wave	133
5.12	Transistor to driving delay	134
5.13	Client side activity diagram	135
5.14	Client side activity diagram	136
5.15	Script A	137
5.16	Script B	138
5.17	Script C	139
5.18	Script D	140
5.19	Script ABCD	141
5.20	Script E	143
5.21	Script EFG	144
5.22	Memory and thread monitoring	147

# LIST OF ABBREAVIATION

JVM	-	Java virtual machine	
API	-	Application programming interface	
IEC	-	International electrotechnical commission	
PLC	-	Programmable logic controllers	
PC	-	Personal computer	
HMI	-	Human machine interface	
SCADA	-	Supervisory control and data acquisition	
MTU	-	Master terminal unit	
RTU	-	Remote terminal units	
DCS	-	Distributed control system	
DAQ	-	Data acquisition	
LAN	-	Local area network	
IDE	-	Integrated Development Environment	
RF	-	Radio frequency	
ЛТ	-	Just in time	
OOP	-	Object-oriented programming	
OS	-	Operating system	
GUI	-	graphical user interface	

RMI	-	Remote method invocation
JME	-	Java micro edition
RTOS	-	Real time operating system
JEE	-	Java enterprise edition
RISC	-	Reduced instruction set computing
JCA	-	Java cryptography Architecture
JCE	-	Java cryptography Extension
AES	-	Advanced encryption standard
IP	- 0	Internet protocol
ТСР	-	Transmission control protocol
HTTP	-	Hypertext transfer protocol
IDC	-	International data cooperation
OSI	-	Open system interconnection
QoS	-	Quality of service

C Universiti Teknikal Malaysia Melaka

# TABLE OF CONTENT

			PAGE
ABS	TRACT	2	iv.
ABS	TRAK		v.
ACK	NOWL	EDGEMENT	vi.
LIST	Г OF ТА	ABLES	vii.
LIST	r of fi	GURES	viii.
LIST	Г OF AI	BBREVIATIONS	xi.
TAB	LE OF	CONTENT	xiii.
1.	INTI	RODUCTION	1
	1.1	Overview	1
	1.2	Problem statement	1
	1.3	Research objective	3
	1.4	Scopes of Research	4
	1.5	Research Methodology	5
	1.6	Thesis Contribution	6
	1.7	Thesis Outline	9
2.	LITH	ERATURE REVIEW	12
	2.1	Overview	12

2.2	Significance of Smartphone	13
2.3	Research question of Java	14
2.4	Ideas and approaches	14
2.5	Significance and benefits of Java	16
2.6	Capabilities of Java	21
2.7	Comparison of Java and other languages	23
2.8	Java disadvantages	26
2.9	Technology trends	27
2.10	Java trends	28
2.11	Java development environment and Configuration	29
2.12	Architecture and Platform	33
2.13	Java Embedded Technologies	35
2.14	Multithreading	35
2.15	Remote method invocation	37
2.16	Java network processor	40
2.17	Java virtual machine implementation debugger	47
2.18	Hibernate framework	49
RESE	ARCH METHODOLOGY AND MODELLING DESIGN	53
3.1	Innovated Design	53
3.2	Front Component Framework Design Technique	54
	3.2.1 Introduction	55
	3.2.2 Callback method	56
	3.2.3 Front component lifecycle	57
	3.2.4 Lifecycle callback	60

3.

	3.2.5	Task and backstack	66
	3.2.6	Method Summary API	70
3.3	Client	Component Framework Design Technique	72
	3.3.1	Introduction	73
	3.3.2	Request execution	73
	3.3.3	HTTP request message	73
	3.3.4	Socket connection	74
	3.3.5	Message headers	77
	3.3.6	Message entity	77
	3.3.7	Using message entity	78
	3.3.8	HTTP transport safety	79
	3.3.9	Idempotent methods	79
	3.3.10	Aborting requests	80
	3.3.11	Connection Persistence	80
	3.3.12	HTTP connection routing	81
	3.3.13	Method Summary API	82
3.4	Server	Component Framework Design Technique	88
	3.4.1	Introduction	89
	3.4.2	Web container	90
	3.4.3	Web server and server machine	91
	3.4.4	Metadata flow	91
	3.4.5	Life cycle	95
	3.4.6	Initialization	96
	3.4.7	Service	97
	3.4.8	Destruction	97

	3.4.9 Process flow	98
	3.4.10 Method Summary API	99
INTE	GRATION AND RESULT DISCUSSION	105
4.1	Overview	105
4.2	Innovated front component	105
4.3	Innovated client component	106
4.4	Innovated server component	106
4.5	Paradigm	107
4.6	System architecture	110

4.

4.7	Handling multi request	112
4.8	Idempotent request	113

4.9	Debugging	113
4.10	Performance tuning	117
4.11	Design principle	119

122

5.	<b>OUTPUT ANALYSIS</b>	AND EVALUATION	

5.1	Overview	122
5.2	Abstraction design	122
5.3	Demonstration and implementation	124
5.4	Motor control system	130
5.5	Lighting control system	135
5.6	Magnetic and sound control system	139
5.7	Driving the relay	140
5.8	Activity and flow diagram	142

	5.9	Script debugging and testing	143		
	5.10	Runtime memory and threads monitoring	152		
6.	CON	CLUSION AND RECOMMENDATION	156		
	6.1	Overview	156		
	6.2	Conclusion	156		
	6.3	Application and usage	157		
	6.4	Trends	158		
	6.5	Weakness	159		
	6.6	Future works	160		
BIBLI	BIBLIOGRAPHY AND REFERENCES				
APPENDIX A					
APPENDIX B			169		
APPENDIX C					

## **CHAPTER 1**

#### INTRODUCTION

#### 1.1 Overview

This chapter discusses the whole thesis briefly as an introduction while sets the scene, research context, problem context and establishes why it is an issue worthy of research. The research question, aims, objectives, scope and context of the research were clearly identified and stated throughout the chapter. At the end of this chapter, each chapter are outlined briefly with introduction, literature review, research methodology and modeling design, integration and result discussion, output analysis and discussion, conclusion and recommendation.

#### 1.2 Problem statements

Currently, there are limited smartphone based solutions for real time distributed control system especially in industrial automation domain. User is needed on site to monitor and control the system. Smartphone controlling regardless of any location is more practical and convenient, thus it is a perfect candidate for automation control solution. Currently, there are also limited Java based solutions for real time distributed control system. Programming language that is used in industrial do not have garbage collector and is platform specific (Ann Wollrath, Jim Waldo, Roger Riggs, 2012). These platform do not have any automatically memory optimizing or memory management

1

mechanism, thus, this will cause redundant and excessive memory allocation in the memory storage. When hardware is updated or replaced by a newer hardware, the entire binary software needed to be changed and modified. Software that has been written is not able to run on different hardware platform without recompiling the source code. Besides that, the same code that is executed on machine is not able to be executed on computer for any hardware simulation or testing. Machine running code is not compatible with computer running code, thus, different types of source code are needed for computer simulation and real time platform respectively. Besides that, code patching and modification require stopping of machine which then will affect the production yields. User is not able to write a real-time dynamic code to diagnose hardware failures in case of emergency or faulty. Users are usually forced into purchasing one brand of hardware or software from specific manufactures due to the platform specific features of programming style that used in industrial. The machine running codes are all tightlycoupled where a small little change of software or hardware required the modification of whole system or even sometime the need of integrating parts that will burden the performance.

The current version (Java 6 update 26) of the Java includes a garbage collector which will dislocate any unused memory automatically and platform independent ability (John Ferguson Smart, 2008). Industrial automation can use Java to produce systems that perform more reliably, improve programmer productivity, time efficiency and even provide binary compatibility across hardware platforms. Using Java, when comes to hardware simulation, the same application class files that run on the target machine or PLC can also execute on a desktop computer for development and debugging due to its platform independent ability. Once the code is completed, the class files can be executed directly on the target hardware for final testing and quality assurance. It simplifies remote monitoring and controlling in several ways where its internetworking based language allows it to run on any web browser or any other networking platform. In the event of a hardware fault, a developer could also write a dynamic Java classes that monitor a set of sensors on the hardware for maintenance and troubleshooting. Not only can dynamic troubleshoot the behavior of the hardware, but it can also be used to patch existing code or add additional functions without stoppage or modification of machine. The ability to change a single source file, recompile it and then dynamically download only the changed class file and link it with a running JVM provides added flexibility for software maintenance. When that technique is used, software in an entire factory can be upgraded remotely, thereby avoiding costly on-site maintenance requests. Hardware independence ability allows developers to simulate the run-time environment on a desktop computer, greatly reducing development time. Another benefit is that debugging is required only on a single version of source code, once for all platforms. This reduces the project overhead and maintenance by greatly simplifying project build configurations, reducing coding errors and simplifying bug tracking and change requests. Users will no longer be forced into purchasing one brand of hardware or software as successful systems can operate on a multiplicity of platforms (hardware and operating system).

#### 1.3 Research Objectives

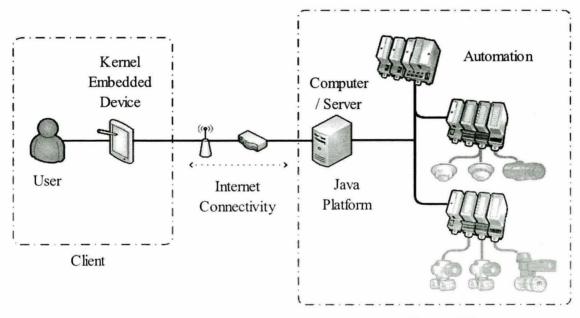
The main goal of the research presented in this thesis is to investigate issues and approach pertinent to integration of Java, Smartphone and hardware and research implementation issues associated with their development. The thesis has the following objectives:

- Extend and adapt Java platform into real time distributed control application environment with Java features inherited.
- Design three platform independent frameworks of component that implement internet protocol which provide interaction over client server paradigm.
- Integrate all frameworks and hardware over an application which allows access to automation control via smartphone over wireless internet network.

## 1.4 Scopes of Research

Java is a programming language and is typically compiled to byte code (class file) that can run on Java Virtual Machine regardless of computer architecture with platform independent ability. Java was used to create a platform independent framework of classes that implement internet protocol which provide interaction over client server paradigm. These frameworks are library or class file that are compatible with any computer regardless any type of operating system. A computer was served as a centralized server before connecting to other automation devices. Server side framework that used to establish connection between client and automation was compiled into a server. Client side framework that used to control and manipulates the system was developed and compiled into the Kernel of the smartphone. Controlling devices are not limited to only smartphone, but as long as targeted devices has a build in operating system, then are compatible with the software that created in this research. In the demonstration or in this research, only one smartphone and one server were used for modeling. These frameworks and software was then integrated over the automation system which allowed access to automation control via smartphone over wireless internet network.

At the end, there appeared only two simplified parts which are client side and server side. A custom circuit for mimicking industrial automation part has been developed in this project for demonstration. Devices or peripherals such as motor, lighting, sound wave and signal related system consists of both analogue and digital signal was integrated in this project. Through this approach, kernel embedded device and Java platform's extendibility and adaption in different environment were proven successfully. Java features that inherited and utilized in this research are limited to object oriented, multithreaded, platform independence, automatic memory allocation, distributed and dynamic networking. The figure 1.1 below illustrates all overall integrated system.



Server Side

Figure 1.1 Simplified Single System Overview

## 1.5 Research Methodology

A research methodology is developed in order to achieve the research objectives. Initially a general flow was decided on how the user steps through that feature and determine the content and behavior that's needed for user to utilize this framework. For each feature, high-level task flow was diagrammed to map the necessary steps a user must take in order to complete a controlling task. As the flows are discussed among the team, details are identified and added to the diagram to show decision points, subtasks, etc. Once the team has a good idea of that features workflow and the necessary data for each step. The screen flow let the development know how many unique functionalities need to be created and also begin to identify some User Interface and controlling behaviors that helped to build consistency in the workflow, such as: certain types of errors send the user back to the originating page, or, search results of one can skip the list results page and send users to the details page. A good wireframes are invaluable in identifying common User Experience design patterns at both the page and widget level. From this information, our research team created a library of framework for reusable designs for any application, noting the appropriate interaction information for the behavior layer. All phases which include literature review, analysis, design, model development, validation, and simulation were used in order to achieve the research objective. Any future enhancements can then be plugged into the flow and evaluated for usability and consistency.

## 1.6 Thesis Contribution

One of the primary contributions of this research is for users that use this framework, This framework allow users to integrate the internetworking environment

with machine running devices and its ability to adapt on different hardware platforms without recompiling the source code. For the industrial automation industry, this has several benefits. First, as hardware is updated and newer hardware controllers replace older ones, no change in the binary software is necessarily required. For example, if a PLC that fully utilizes a 40-MHz Intel processor is upgraded to a 100-MHz Power PC processor, then the same binary class files can be used for both controllers, assuming that hardware ports and external interfaces are programmed for portability. Second, hardware independence allows developers to simulate the run-time environment on a desktop computer, greatly reducing development time. With the simple addition of lowlevel hardware simulation, the same application class files that run on the target PLC can also execute on a desktop computer for development and debugging. Once the code is completed, the class files can be executed on the target hardware for final testing and quality assurance. This saves valuable time during the development process, since standard software development environments may be used. Familiar debugging tools and development environments that are not tied to specific hardware platforms are easier for developers to learn and use; also, testing does not require the constant downloading and setup of the target PLC. Another contribution is that debugging is required only on a single version of source code, once for all platforms. This reduces the project overhead and maintenance by greatly simplifying project build configurations, reducing coding errors and simplifying bug tracking and change requests once this framework was implemented in user environment. The idea of remote controlling via smartphone has already proved to be effective in the computer server market, where high-end servers are remotely managed through a Web interface. Developers could also use dynamic downloading to diagnose hardware failures. For example, in the event of a hardware fault, a developer could write source code that monitor a set of sensors on the hardware,