World Academy of Science, Engineering and Technology
International Journal of Industrial and Manufacturing Engineering
Vol:5, No:1, 2011

Industrial Waste Monitoring

Khairuddin Bin Osman, Ngo Boon Kiat, A. Hamid Bin hamidon, Khairul Azha Bin A. Aziz, Hazli Rafis Bin Abdul Rahman, Mazran Bin Esro

***Abstract*—**Conventional industrial monitoring systems are tedious, inefficient and the at times integrity of the data is unreliable. The objective of this system is to monitor industrial processes specifically the fluid level which will measure the instantaneous fluid level parameter and respond by text messaging the exact value of the parameter to the user when being enquired by a privileged access user. The development of the embedded program code and the circuit for fluid level measuring are discussed as well. Suggestions for future implementations and efficient remote monitoring works are included.

***Keywords*—**Industrial monitoring system, text messaging, embedded programming.

I. Introduction

MONITORING on industrial waste water discharge had been implemented across the country since decades but is usually confine to site. Monitoring and controlling industrial process maybe a tedious task where a person must be employed on site in order to monitor an industrial process which is a waste of money and time should there be no problem on site. Environmental Quality Act, 1974 and the Environmental Quality (Sewage and Industrial Effluents) Regulations, 1979 [1], requires all industries with known point source of waste water discharge to install, monitor and report flow measurement of wastewater discharges from an industrial outlet. Such method of monitoring is a time-consuming task, inefficient, subjected to fraudulence and centralize monitoring is almost impossible at times due to the site locality and limited resources of personnel present. Higher officials are unable to acquire first hand data but rather have to go through numerous unreliable intermediate channels.

II. Literature Review

A. Embedded Systems and Computing

In November 2005, Andrew David Moss [2] developed a program transformation tools in the analysis and compilation of programs for embedded systems to aid the programmer in understanding and controlling the effects towards software precision and timing and therefore reduces the complexity of the problem. With the advent of system level integration (SLI)—the next level of integration beyond Very Large System Integration (VLSI)—and system-on-chip (SOC) capabilities, the computer industry's focus is shifting from personal to embedded computing.

The opportunities, needs, and constraints of this emerging trend will lead to significantly different computer architectures at both the
system and processor levels as well as a rich diversity of off-the-shelf (OTS) and custom designs.

Driven by the accelerated pace of semiconductor integration during the past three decades, the computer industry has steadily moved from mainframes and minicomputers to workstations and PCs. In accordance with a corollary of Moore's law, computing power becomes half as expensive every 18 to 24 months. Over a decade, this reduces the cost by a factor of 30 to 100, making computing affordable to an exponentially larger number of users and dramatically changing the key applications of this computing power. Manufacturers have for several years incorporated embedded computers in so-called smart products such as video games, DVD players, televisions, printers, scanners, cellular phones, and robotic vacuum cleaners. Using embedded computers in devices that previously relied on analog circuitry such as digital cameras, digital camcorders, digital personal recorders, internet radios, and internet telephones provides revolutionary performance and functionality that analog designs could not achieve [3]. Any computer architecture must balance the latest technological opportunities with product, market, and application requirements that together determine three important features of embedded computing architecture: specialization, customization, and automation. Specialization increases the performance and reduces the manufacturing cost of embedded computer systems. Customization permits specialization when no adequately specialized OTS product is available. Automation reduces the design costs incurred by customization.

B. Teltonika T-Box N12R

T-BoxN12R was designed for M2M (machine-to-machine) applications or other wireless solutions. Integrated Nokia 12 GSM module enables flexible wireless communication over GSM network connecting external hardware (controllers for Real-Time operation) to RS-232 interface. All these features enable to use T-BoxN12R in wide range of applications.

Khairuddin Bin Osman is with Faculty of Electronic Engineering and Computer Engineering, Universiti Teknikal, , Malaysia.(E-mail:khairuddin.osman@utem.edu.my)

Fig. 1 Teltonika T-BoxN12R

It is an open architecture device, which is fully programmable, the device can be adapted to our own needs by programming or writing your own JAVA IMlet. The GSM module has up to 7 digital inputs, up to 8 digital outputs, 3 analog inputs and 2 usable RS232 ports, which could be used for performing tasks on remote objects, such as monitoring air or water temperature, humidity, or switching lamps, motors etc.

C. Principles of Ultrasonic sensors

Ultrasonic sensors transmit ultrasonic waves from its sensor head and receive ultrasonic waves reflected back from an object, therefore an ultrasonic sensor generally would involve the use of a transmitter and a receiver.

By measuring the length of time from the transmission to reception of the sonic wave, it measures the distance of the object as shown in Fig. 2.

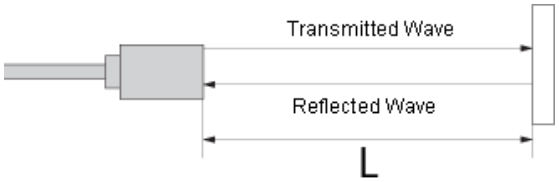

Fig. 2 Diagram of calculating the length L of the distance from the transmitted and received wave.

III. Methodology

A. Wireless Messaging API

The Wireless Messaging API was used to send and receive short messages. The Wireless Messaging API specification was obtained at http://jcp.org/aboutJAVA/communityprocess/final/jsr120/index2.html and the JAVAdocs are available at http://JAVA.sun.com/products/wma/.

The application would first obtain an instance of the MessageConnection through the Connector class. The URL was then passed to the JAVAx.microedition.io.Connector.open() method that identifies the protocol to be used (SMS), the phone number and/or port of the target. Valid URLs are as follows:

1) sms://+0123xxxx
2) sms://+0123xxxx:5678
3) sms://:5678

B. Openning a connection

As in any communication that uses IMlets, the time period from starting the Nokia 12 module to registering into the network can vary in different networks. Another way to wait for the network registration is simply to try and open a connection, catch the raised exception, and try again until the connection succeeds.

C. Sending a Text Message

To send a message, the MessageConnection.newMessage() method was used to create an empty message, set its payload (text or binary data to be sent), and then the MessageConnection.send() method was invoked.

D. Receiving a Text Message

There are two methods in receiving a text message. Either incoming messages are channeled through a receiving port that listens to incoming text messages using the Wireless Messaging API, or the device can read the received message from the SIM card and then deletes it after reading it using the Embedded Terminal Module.

1) Using the Wireless Messaging API
   The server connection must be opened with the port number, but without the phone number. After opening the connection, the MessageConnection.receive() method is called. This returns the next available message to the specified port. If there is no message available, the method blocks until a message arrives or a different thread closes the connection.

2) Using the Embedded Terminal Module
   The previous method of receiving messages via port number will set a limitation for the amount of SMS received and would not be efficient as it receives messages by listening through a port number for incoming text messages. Since local Malaysian telecommunication networks do not disclose the port number being used for subscribers to receive text messages, the Wireless Messaging API method cannot be used.

Another alternative method can be used rather than to receive incoming text messages through a port number, the Embedded Terminal or (ET Module) shall be used from the package com.nokia.m2m.orb.idl.terminal.ETPackage. This method reads the latest received text message from the SIM card and then deletes it after the instruction has been executed

E. I/O Control

Some of the methods are only supported by the real Nokia 12 GSM module, not by the Nokia 12 IMP 1.0 Concept Simulator. Methods not supported by the Nokia 12 IMP 1.0 Concept Simulator are defined in this interface for compatibility but they will not implement any functionality.

1) Using the I/O Control API
The IOControl API is a Nokia proprietary API. IMlets using this API cannot be used with M2M devices from other manufacturers. The IOControl class is used to control the input and output pins of the Nokia 12 module. The number of available pins depends on the current port settings of the Nokia 12 module. The Nokia 12 Configurator is a useful tool for checking the available pins in the current configuration.

2) Using the Embedded Terminal Module
This service is used for controlling and observing the I/O pins of the Nokia 12 module. Module ORB servant implementing the IOControlOperations interface. Reference to this servant can be obtained by using the object key ORB/OA/IDL:IOModule/IOControl:1.0.

F. Simulating and Loading the programmed JAVA IMlet

The Teltonika T-Box N12R uses the Nokia 12 IMP 1.0 Concept Simulator to simulate programmed JAVA IMlets to observe whether the desired result performs the correct behavior and simulates it through the Nokia 12i module as seen in Fig. 3.

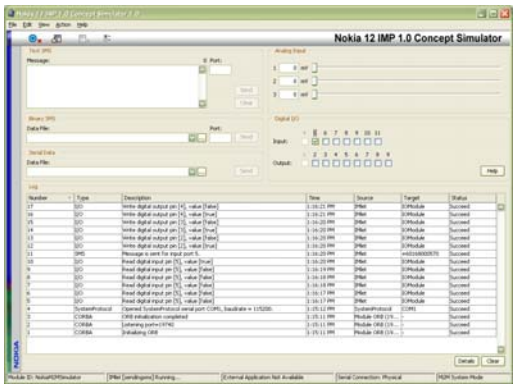

Fig. 3 Nokia 12 IMP 1.0 Concept Simulator

Once the simulated results were successful, the .java extension source code file was then packaged into .jad and .jar extensions to be loaded into the Nokia 12 Configurator as seen in Fig. 4.

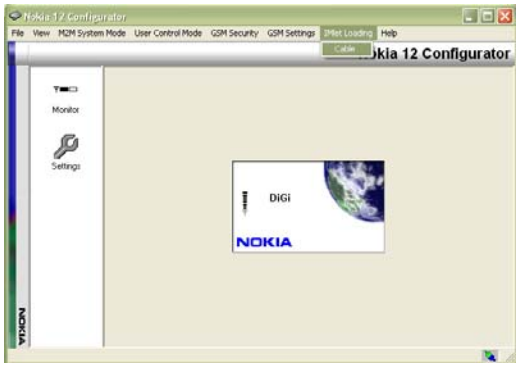

Fig. 4 Nokia Configurator

Then using the Nokia Configurator, the programmed JAVA IMlet is then being loaded into the Teltonika T-Box N12R and then executed.

IV. Results and Analysis

A. Receiving Text Message

For this aspect of the project, the Teltonika T-Box N12R was programmed to read the received text messages from the SIM card, and then delete the received messages from the SIM card using the Embedded Terminal module and then decrement the message count. The I/O Control API can be used to receive incoming text messages by which the received text message will not be stored into the SIM card but rather would be directly "fed" into the Nokia12i module through a listening port number. However, this listening port varies according to the telecommunication networks and the local telecommunication network companies were reluctant and unwilling to disclose such details to the public even though a strong request was suggested for industrial works, therefore an alternative method would have to be implemented. Instead, another method is by receiving the text messages like normal text messages does being stored in SIM card and using the Embedded Terminal to extract the received text message directly from the SIM card. This would ensure that every telecommunication network used by the Teltonika T-Box N12R would still be adaptable to receive text message.

The device reads the text message and the compares the text message, once the content of the text message matches, it will set the digital output pin of the Teltonika T-Box N12R number 5 as high as an indicator.

Fig. 5 Receiving text message to trigger digital output

B. Sending Text Messages

The sending message program code applies to all telecommunication network protocols as it needs to define a connection object that will open the connection to the Telecommunication Network. The message sent will be set with payload text along with the message before sending in order to enable protocols at the receiving end to decode or execute specific modes before actually sending a text message to the receiver.

The device was tested by programming a trigger at digital input pin number 5 to send out a message to a predefined recipient number as follows.

Fig. 6 Triggered digital input no 5 on the Teltonika T-Box N12R

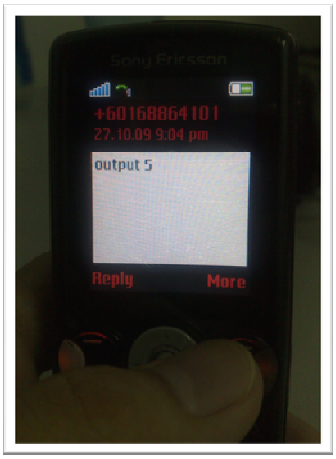

Fig. 7 The sent text messages send out by the Teltonika T-BoxN1R received by the predefined user

C. Analog Voltage to Parameter Conversion

Once the receiving and sending text message through the Teltonika T-Box N12R was successful, the analog voltage conversion was conducted.

The Teltonika T-BoxN12R casing is able to receive 0-12V of analog input with 0-2809mV internally for the Nokia 12i module, therefore programming works were done for analog voltage input with an integer value range of 0 to 2809. However, conversion from analog voltage into measured parameters requires the division operation which means floating-point will be involved in the process but embedded JAVA does not support floating-point formats such as double type or float type in String format. One of the methods to resolve this problem for the time being was using multiplication to multiply the ratio with a factor (x1000), then divide the final answer back with that factor. Value cut-off without round-up or round-down would exist but it will have an accuracy of 4 digits cut-off.

D. Power Failure Notification Alert

This monitoring system will incorporate a power failure notification alert function whenever the power source of the industrial outlet is absent or experienced a power failure. In the event of a power failure, the Teltonika T-Box N12R will be programmed to send out a text message to a pre-defined user's number but will confirm the existence of a power failure after about 5 seconds of delay time to prevent from misleading notification alerts to the user.

The program code in shows the "if" statement loop used to detect the power failure of the digital input pins for a low or "0" state. Whenever the input is low, the process is blocked by a delay of 3000 milliseconds equivalent to 3 seconds, and then enters another nested "if" loop to proceed with sending a power failure notification alert whenever the power failure is confirmed.

E. Parameter Value Conversion

The parameter value in absolute integer value would be converted into decimal fractions (e.g. 4510mm >> 4.51m) through arithmetic operational manipulating methods to separate the integer part on the left and the decimal fraction on the right separated by a period (.) as seen in Fig. 8.

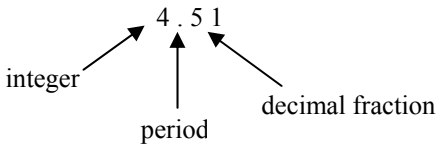

Fig. 8 Representation of integer, period and decimal fraction

Once the parameter value was calculated, using the above example, 4510mm. The value was divided by 1000 and then stored as an integer in variable "numbers". Due to the characteristics of embedded systems of not being able to perform arithmetic operations in decimal fractions, dividing 4510 by 1000 would yield 4 instead of the conventional answer 4.51. Then the previous value of 4510 was then used again to deduct the product of ("numbers" x 1000) and then answer divided by 10 to obtain the absolute integer value of decimal fraction after the period, then the answer was then stored in the variable "fractions" .

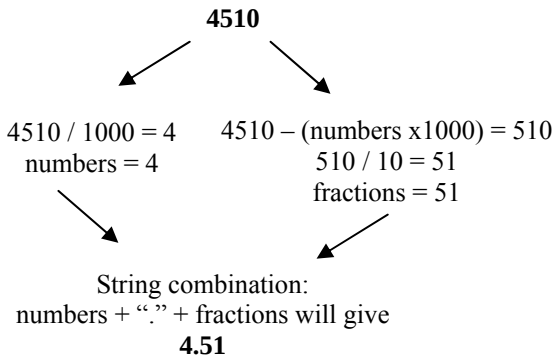

Fig. 9 Representation of integer, period and decimal fraction

F. Embedded System Response Analysis

Embedded system architecture has a tradeoff in performance between flexibility, versatility and speed of processing speed. An analysis between the relationship of the response time of the embedded architecture with the length and complications of the program coding was being performed using indicators included within the program. The first programmed line is the code to set both digital output pins 2 and 9 to low. The second line of code to start the mark of the code for setting digital output pin number 2 to high is at the third line and the fourth line indicates the code to set the digital output pin number 9 to low at the end of the main function.

The response will be measured by introducing two sets of program codes, one with function calls from the main function linking to multiple function calls to address a public function method, and another set of code that executes the program code directly within one main function without the function calls. As the program process proceeds within the "while" loop, the first indicator was introduced by setting the digital output of pin number 2 high at the beginning of the main function call, then a code was added to set the digital pin number 9 to high at the end of the main program function. The time taken from the start whenever digital output pin number 2 was high marks the start of the process duration and whenever digital output pin number 9 is high, it marks the end of the process duration. A time frame within both points of indication marks the duration of response time of the embedded system architecture for the Teltonika T-Box N12R. The response analysis will be using two sets of coding.

1) Actual program code with multiple functions calls from main function.
2) Modified tested program code without function calls.

Note that the program code is the actual program code for the project, however the program nature for the analysis purpose would be in the pending or standby state of the response without power failure or any user query/response activity (sending or receiving message).

G. Retrieval of Analysis Result of Process Duration

The program used will be first loaded into the module, then the module rebooted. The duration between the two digital outputs will start to be recorded after a short period of time for stability of the system.

From the analysis results in Table 1 and Table 2, the program with only a single "if…else" statement within a single "while" loop without function calls produced a faster processing speed. Significantly a deduction can be made regarding this issue that embedded programming architecture would perform much faster without complicated programs, in this case without the need for multiple function calls. Function calls may seem convenient for a developer or a programmer to develop a system, however to ensure smooth operational performance and efficiency in processing speed, simpler programming practices is best preferred.

A graph can then be plotted as shown based on the analysis of the duration of the embedded architecture's processing speed versus the complications of the program code.

TABLE I
PROGRAM CODE WITH FUNCTION CALL

Program code with function call		
Initial (Pin 2)	End (Pin 9)	Duration
2s	6.9s	4.9s
2s	6.0s	4.0s
2s	6.4s	4.4s
2s	6.6s	4.6s
2s	6.4s	4.4s
Average		**4.46s**

TABLE II
PROGRAM CODE WITHOUT FUNCTION CALL

Program code without function call		
Initial(Pin 2)	End (Pin 9)	Duration
2s	5.7	3.7s
2s	5.4	3.4s
2s	5.6	3.6s
2s	5.4	3.4s
2s	5.8	3.8s
Average		**3.58s**

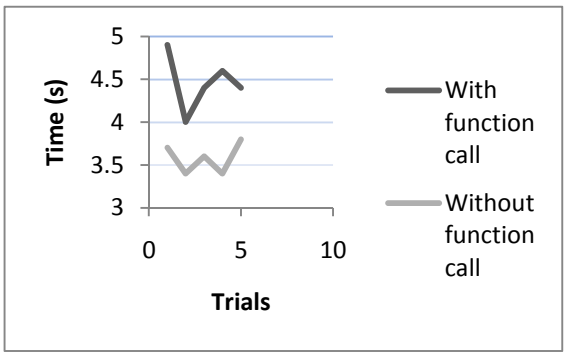

Fig. 10 Processing Speed vs Programming Code

H. Accuracy Analysis

The project was then tested and the results analyzed for its accuracy in the sense of the programming in arithmetic operations to calculate the value of an external parameter (tank level) from the analog voltage input from the practical results and the simulated results.

I. Simulation Results

In order to give a fundamental idea of the project, the table below shows the correlation between the analog input voltage and the parameter value (tank level) obtained from a theoretical simulation for the output result. The fixed parameters value for the tank level is minimum 0 meter to 8 meters maximum and analog voltage minimum is 0V to maximum 12V.

TABLE III
CORRELATION BETWEEN ANALOG VOLTAGE INPUT WITH FLUID LEVEL (SIMULATION)

No	Input analog voltage range (V)	Parameter value, fluid level (m)
1	0.0	0.00
2	2.0	1.33
3	4.0	2.67
4	6.0	4.00
5	8.0	5.33
6	10.0	6.67
7	12.0	8.00

A graph was then plotted based on the data in Table 3 as shown in Fig. 11.

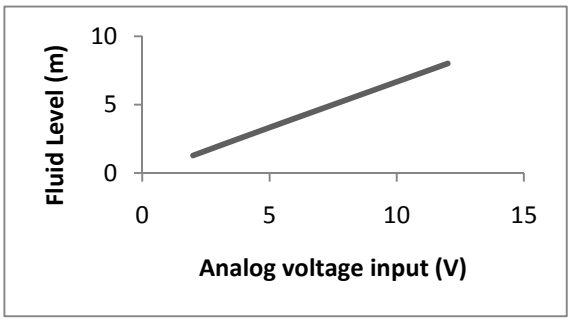

Fig. 11 Analog Voltage Input vs Fluid Level Height (simulation)

J. Actual Practical Results

Practical trials were tested by adjusting the analog voltage input as accurately as possible to the data as in Table II. A text message was then sent to the Teltonika T-Box N12R to query for the current fluid level of the tank and then a text message was responded and sent out from the module to the user with the parameter value of the measured fluid level.

The data received were then tabulated and recorded in Table 4.

TABLE IV
CORRELATION BETWEEN ANALOG VOLTAGE INPUT AND FLUID LEVEL (PRACTICAL)

No	Input analog voltage range (V)	Parameter value, fluid level (m)
1	0.000	0.00
2	2.013	1.42
3	4.010	2.80
4	6.010	4.24
5	8.090	5.65
6	9.990	6.96
7	12.08	8.00

A graph was then plotted based on the data in Table 4 as shown in Fig. 12.

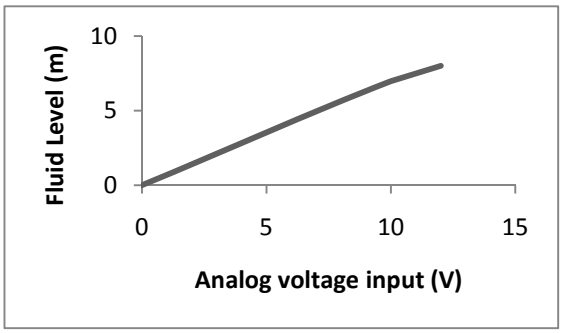

Fig. 12 Analog Voltage Input vs Fluid Level Height (practical)

K. Analysis of Simulation and Practical Results

From Fig. 11 and Fig. 12, the difference of both simulation and practical results are both linear, however as observed from the graph the parameter value pattern for practical results deviates from the ideal simulated results after an analog voltage input of 10V onwards.

From Table 5 and Fig. 13, the deviation of the simulated results and the practical results differs from the ideal simulated results slightly from the beginning, however as the analog input voltage increases to 6V, the ideal height of the practical results for the fluid level has the largest deviation from the ideal simulated results at 10.5% deviation. However, the deviation gradually decreases as the analog voltage input approaches the maximum value.

In comparison, the ideally simulated results exhibits perfect linearity and however the practical results shows slight deviation as seen from the graph in Fig. 13.

TABLE V
COMPARISON BETWEEN SIMULATION AND PRACTICAL RESULTS

No	Simulation parameter value , fluid level (m)	Practical parameter value, fluid level (m)	Deviation from ideal simulated results.	Percentage deviation (%)
1	0.00	0.00	0	0.00%
2	1.33	1.42	0.09	6.77%
3	2.67	2.80	0.13	4.87%
4	4.00	4.24	0.42	10.5%
5	5.33	5.65	0.32	6.00%
6	6.67	6.96	0.29	4.35%
7	8.00	8.00	0	0%

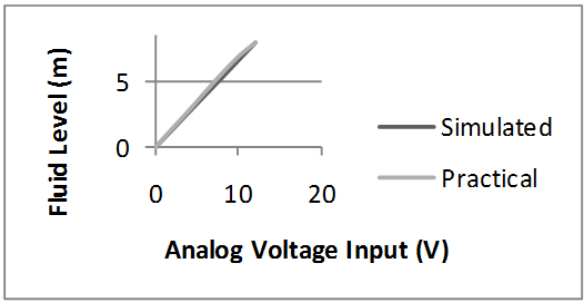

Fig. 13 Comparison between simulation and practical results

This analysis shows the limitations of the embedded architecture in the accuracy and integrity of the measured values. This deviation occurs as the Nokia 12i module inside the Teltonika T-Box N12R accepts analog input values of millivolts up to 2809mV relative to 12V from the T-BoxN12R case. However, the Teltonika T-Box N12R accepts analog voltage input range up to 12V. This results in another layer of filtering from the accuracy of the first level raw analog voltage input of 0 - 12V and being converted into the range of 0mV - 2809mV instead.

V. Conclusion

This remote monitoring via text messaging applied towards industrial use by far is the most efficient method of monitoring that saves man power and does not require 24 hours human monitoring. Data is available as a first hand direct approach where the integrity of the data is maintained.

The architecture of an embedded system is fairly significant and favorable in resolving challenges faced when dealing with new systems. The most common of these challenges include:

1) Defining and capturing the design of a system
2) Cost limitations
3) Determining a system's integrity, such as reliability and safety
4) Working within the confines of available elemental functionality (i.e., processing power, memory, battery life, etc.)
5) Marketability and salability
6) Deterministic requirements

Among others, this project can be implemented for monitoring for:

1) Factory machines.
2) Municipal water & wastewater treatment plants.
3) Agricultural irrigation.
4) Heating, cooling & refrigeration equipment.
5) Process monitoring & control.
6) Condition monitoring & control.
7) Mining & power generation plant and etc.

Moreover, the Teltonika T-BoxN12R also supports GPRS and 3G applications with the Nokia 12i module, therefore future applications can be applied with a Web-based remote monitoring where parameter data can be monitored and recorded into an online database server for remote accessibility anywhere with the availability of Internet access.

Acknowledgment

The authors gratefully acknowledge of Universiti Teknikal Malaysia Melaka (UTeM) for her encouragement and financial support.

References

[1] *Environmental Quality Act 1974 and Regulations Malaysia*. Retrieved from http://openlibrary.org/b/OL22508292M/Environmental_Quality_Act_1974_and_regulations.

[2] Moss, A.D., 2005, K4e*, Program transformation of embedded systems*, Ph.D., Bristol, 56-5121 (BL: DXN103086)

[3] Prof. Dr. Friedel Hoβfeld, Super-Computer Evolution – Along Moore's Law and Beyond. Retrieved from http://www.eml-development.de/deutsch/veranstaltungen/kolloquien.php?we_objectID=238

[4] Noergaard T., *Embedded Systems Architecture – A Comprehensive Guide for Engineers and Programmers.*

[5] Wiley J. & Sons, *Mobile Messaging Technologies and Services: SMS, EMS and MMS*.

[6] Mulchandani D., *JAVA for Embedded Systems*. IEEE explore.

[7] A. Weaver, J. Luo, and X. Zhang, *Monitoring and control using the internet and java*, in Proceedings of the 25th Annual Conference of the IEEE Industrial Electronics Society (IECON'99), vol. 3, 1999, pp. 1152–1158.

[8] M. J. Callaghan, J. Harkin, T. M. McGinnity, and L. Maguire, *An internet-based methodology for remotely accesses embedded systems*, in Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, vol. 6, Oct. 2002.

[9] F. Chen and G. Ros¸u. Java-MOP: *A Monitoring Oriented Programming Environment for Java*. In Proceedings of the Eleventh International Conference on Tools and Algorithms for the construction and analysis of systems (TACAS'05), LNCS 3440, pages 546–550. Springer-Verlag, 2005.

[10] J. Ligatti. *Policy Enforcement via ProgramMonitoring*. PhD thesis, Princeton University, Department of Computer Science, 2006.

[11] R. Sekar, C. R. Ramakrishnan, I. V. Ramakrishnan, and S. A. Smolka. Model-Carrying Code (MCC): a new paradigm for mobile-code security. In *Proceedings of the 2001 Workshop on New Security Paradigms, NSPW'01*, pages 23–30, New York, NY, USA, 2001. ACM Press.

[12] Fei Xie, Guowu Yang , Xiaoyu Song, *Component-based hardware/software co-verification for building trustworthy embedded systems*, The Journal of Systems and Software,2007, 80,pp: 643–654

[13] Carlos Eduardo Pereira , Luigi Carro, *Distributed real-time embedded systems: Recent advances, future trends and their impact on manufacturing plant control*, Annual Reviews in Control ,2007,31,pp: 81–92