An Implementation Scheme for the Detection System of RFID Defective Tags Using LabVIEW OOP

Deok-Gil Jung, Min-Po Jung, Hyuk-Gyu Cho and YoungUhg Lho, *Member, KIMICS*

***Abstract*— In this paper, we suggest the object-oriented methodology for the design and implementation scheme for the program development in the application of control and instrumentation such as the detection system of RFID defective tags which needs the embedded programming. We apply the design methodology of UML in the system design phase, and suggest the implementation scheme of LabVIEW programs using LVOOP(LabVIEW Object Oriented Programming) in which make it possible to write the object-oriented programming. We design the class diagram and the sequence diagram using UML, and write the classes of LVOOP from the designed class diagram and the main VI from the sequence diagram, respectively. We show that it is possible to develop the embedded programs such as the RFID application through the implementation example of the detection system of RFID defective tags in this paper. And, we obtain the advantages based on the object-oriented design and implementation using the LVOOP approach such as the development of LabVIEW programs by adding the classes and the concept of object of the object-oriented language to LabVIEW.**

***Index Terms*— Defective Tag Detection, Embedded Programming, LVOOP, RFID Tag, UML**

I. INTRODUCTION

The OOP(Object-Oriented Programming) methodology in the development process of computer software is recently applied and used as the inevitable development methodology in various application fields, and it is more useful especially in the development of enormous and complex engineering system[1]. And, LabVIEW[2], which is the visual programming language, makes it easy to learn the programming, possible to be skilled in a short time, easy to develop the software for the system which constitutes various types of engineering hardware environments, and has the advantages of raising the productivity[3]. Also, as the applications of LabVIEW become more large and complex, it feels keenly the necessity of OOP techniques which apply in the development of LabVIEW programs[4],[5]. Thus, in order to obtain the advantages of object-oriented techniques which are used in the development of computer software, many functionalities related to the OOP support is added to the integration of LabVIEW development environment in version 8.2 of LabVIEW[6]. The implementation methodology of OOP into LabVIEW not only has many advantages of added functionalities such as the inheritance, the native applications of LabVIEW such as the data flow is also useful and has the advantages of successful system development[7],[8].

The detection process of the defective tags in most of Korean domestic RFID manufacturing companies is treated by hand-operated processing after the chip bonding job, and this validation check process depends on the skilled levels of workers, so it has been requested to reduce the time and the cost for manufacturing of RFID tags[9]. Therefore, in this paper, we suggest the design and implementation scheme of the system for detecting of the RFID defective tags after the job of chip bonding.

UML(Universal Modeling Language)[10],[11] is used as a representative modeling technique for the object-oriented analysis and design of computer software. In this paper, we suggest the implementation scheme using LabVIEW Object Oriented Programming(LVOOP)[6],[7] after design for the detection system of RFID defective tags using UML. We construct the class diagram and the sequence diagram using UML, and generate the classes of LVOOP from the designed class diagram and the main VI(Virtual Instrument) from the sequence diagram, respectively. According to this implementation scheme, we show that it is possible to develop the embedded programs such as the RFID application through the implementation example of the detection system of RFID defective tags by only giving the efforts of UML design level. And, we obtain the advantages based on the object-oriented design and implementation using the LVOOP approach such as the development of LabVIEW programs by adding the classes and the concept of object of the object-oriented language to LabVIEW.

Manuscript received December 21, 2010; revised January 17, 2011; accepted January 21, 2011.

Deok-Gil Jung is with the Department of Computer Science, Dongeui University, Busan, 614-714, Korea (Email: dgjung@hangook.ac.kr)

Min-Po Jung and Hyuk-Gyu Cho are with the Department of Cyber Police, YoungSan University, Kyungsangnamdo, 626-790, Korea (Email: {minpo, hgcho3}@ysu.ac.kr)

YoungUhg Lho is with the Department of Computer Education, Silla University, Busan, 617-736, Korea (Email: yulho@silla.ac.kr)

II. DEVELOPMENT SCHEME USING LABVIEW OOP

In this paper, we construct the development phases of LabVIEW program using LabVIEW OOP as depicted in Fig.1.

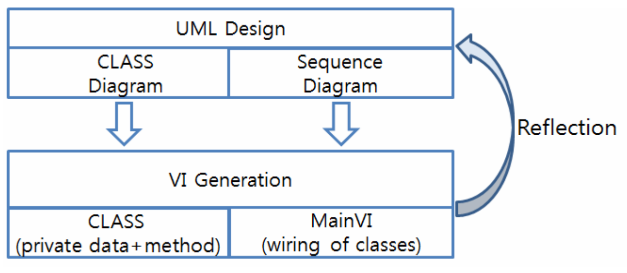


Fig. 1 Development model using LVOOP

A. Design Phase of Classes using UML

By analyzing the system, at first phase, we represent the classes and their relationships into the class diagrams and the sequence diagrams which construct the system using UML. The class diagram shows the structural perspective of the system which represents the structure and relationship among classes, and the sequence diagram shows the scenario concerning the interaction among objects during the execution time.

B. Implementation Phase of LabVIEW Classes using LVOOP

We write the LabVIEW classes using LVOOP, that is, write the private data members and method VIs which construct the LabVIEW class. We write the necessary classes and methods from the class diagrams, main VI from the sequence diagram by integrating the classes.

C. Phase of Class Reflection

When it needs the modifications at the implementation phase of classes, the design phase of classes by UML and the implementation phase of LVOOP classes are iterated. If the modifications happen, it returns back to the design phase of classes by UML and reflects back to the classes, so modifies the design of classes and iterates the implementation process of updates of LVOOP classes accordingly.

III. MODELING THE DETECTION SYSTEM OF DEFECTIVE TAGS USING UML

A. Class Diagram

The major classes of the detection system of RFID defective tags are *frmMain*, *Sensor*, *ImagePanel*, *DBWrapper*, *Options*, *Validation*, and its class diagram is depicted in Fig. 2. The *Sensor* class has 2 subclasses(*Machine_Controller*, *RFID_Reader*), and has the communication function between RFID reader and other devices. The *ImagePanel* class has the function of displaying the information about tag, sensor, and fault information. The *DBWrapper* class has the function to store the tag information and option setting in the system. The *Options* class has the function of option setting such as GUI option, tag property information and communication. The *Validation* class performs the decision activity for defection.

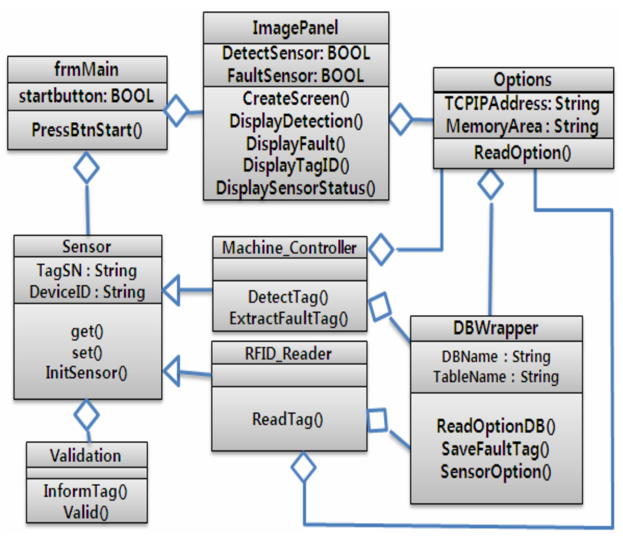


Fig.2 Class diagram

B. Sequence Diagram of Decision for Defection

The sequence diagram which shows the messages and their orders among objects in the detection system of RFID defective tags is depicted in Fig. 3.

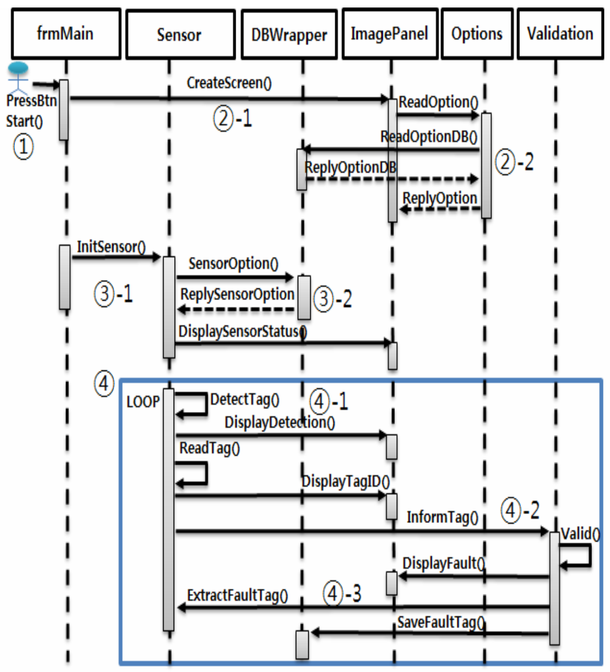


Fig.3 Sequence diagram

① Start of the test

The call to *PressBtnStart()* method to choose one of the hardware or software buttons of Actor start the test for detection of RFID defective tags.

② Primary setting and configuration of the screen for the test

②-1. Call the *CreateScreen()* method of *ImagePanel* object to configure the screen for the detection of defective tags.

②-2. Read the tag type and information of memory from the database and display the information on the screen by calling the *ReadOption()* method of *DBWrapper* object and *ReadOptionDB()* method of *Option* object.

③ Initialization of the sensor

③-1. Try to initialize the sensor by calling the *InitSensor()* method of *Sensor* object from *frmMain* object.

③-2. Initialize the sensor by reading the initial values of the sensor from the call of *SensorOption()* method of *DBWrapper* object, and display the values on the screen by the *DisplaySensorStatus()* method of *ImagePanel* object.

④ Input of tag information from the sensor and decision for defection

④-1. Sense the entry of tag by *DetectTag()* method of *Sensor* object, and display its sensing information on the screen by *DisplayDetection()* method of *ImagePanel* object. Read the tag by *ReadTag()* method of *Sensor* object, and display the value of tag ID on the screen by *DisplayTagID()* method of *ImagePanel* object.

④-2. Decide the defection by *Valid()* method of *Validation* object based on the tag information, and display the information of decided defection on the screen by *DisplayFault()* method of *ImagePanel* object.

④-3. Call the *ExtractFaultTAG()* method of *Sensor* object to extract the defective tag and the *SaveFaultTag()* method of *DBWrapper* object to store the fault information to the database.

IV. LABVIEW PROGRAM USING LVOOP

A. Class Diagram

(1) Object Tree

The object tree for the implementation of the fault control system consists of *frmMain, Sensor, Validation, ImagePanel, Options, DBWrapper* classes, and is depicted in LVOOP project explorer of Fig. 4. *Sensor* class(*Sensor.lvclass*) consists of the private data member(*Sensor.ctl*) and the method VI(*InitSensor.vi*) to represent the interface of the class. Each VI to represent the method is mapped into the concept of sub VI of native LabVIEW.

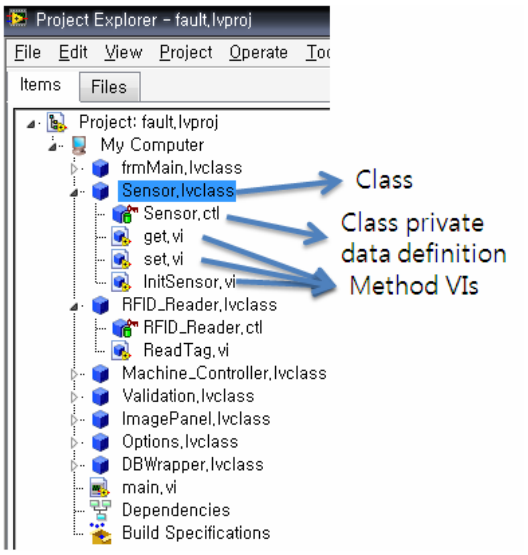


Fig.4 Object tree

(2) Class Hierarchy

The class name of class diagram designed by UML is mapped into each class of LVOOP, respectively. The *RFID_Reader* and *Machine_Controller* classes depicted in the class diagram of Fig. 4 are inherited from the *Sensor* class. The concerning class diagram is depicted in Fig. 5.

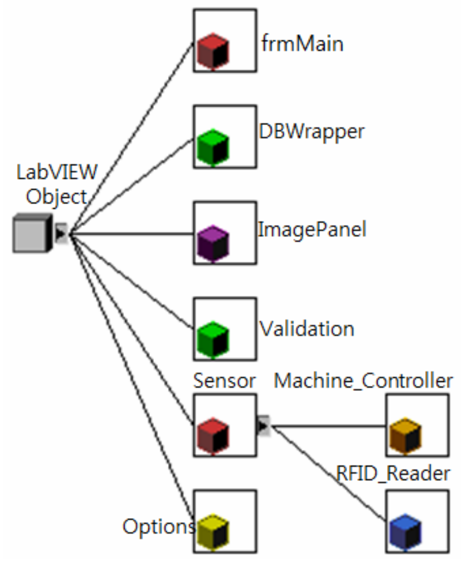


Fig.5 LabVIEW class hierarchy

B. Object-Oriented Programming Using LVOOP

(1) Implementation of Class Private Data

The private data member of LVOOP specifies the wire content. All of classes have just one *.ctl*(control) as shown in Fig. 6. Each *ctl* can define the control of LabVIEW as the data variable.

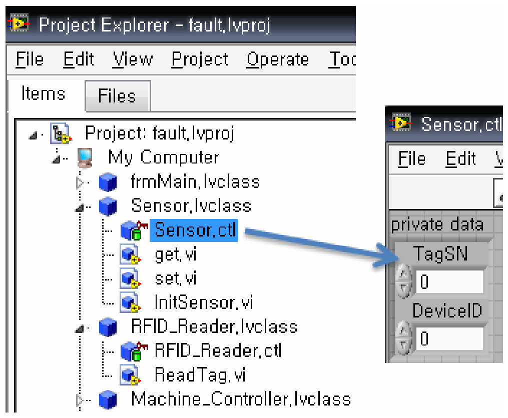


Fig.6 Class private data

(2) Class Method VI

The method VI of class modifies the status of object, and the private data member of class is available only within the method VI. The implementation of *ReadTag()* method of *RFID_Reader* class is depicted in Fig. 7. The *RFID_Reader* class is the subclass of *Sensor* class, and shows that the *TagSN* and *DeviceID* which are the private data members of *Sensor* class are used in the subclass *RFID_Reader*. When the child class accesses the private data member of its parent class, it must use the data only by the method implemented in its parent class. The *set()* method of *Sensor* object assigns the information of the tag(*TagSN*, *DeviceID*) into the corresponding data member respectively, and returns the values by obtaining the private data members using the *get()* method.

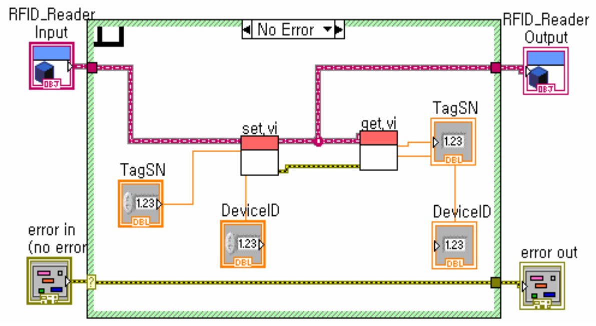


Fig.7 *ReadTag()* method of *RFID_Reader* class

(3) LabVIEW Application Using Classes

The application of LVOOP is executed by calling the method of object. The application shown in Fig. 8 is implemented according to the classes designed in class diagram(Fig. 2) and the exchange of messages among objects and their orders designed in sequence diagram(Fig. 3) using LVOOP.

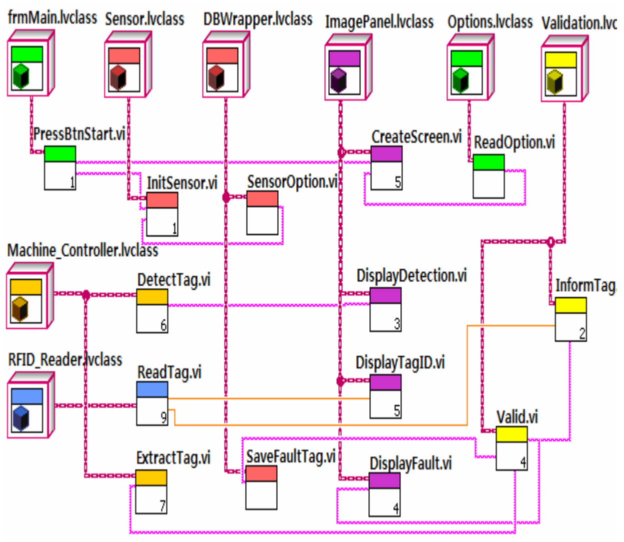


Fig.8 LabVIEW application using classes

V. IMPLEMENTATION RESULT

A. Detection Facility of Defective Tags

Fig. 9 shows the detection facility of RFID defective tags which classify the RFID defective tags. The configuration of the facility is equipped by the Velleman-K8055 interface board[12] which consists of 5 digital inputs and 2 analogue inputs. The RFID reader is tested by the ALR-9000 version of Alient Technology[13].

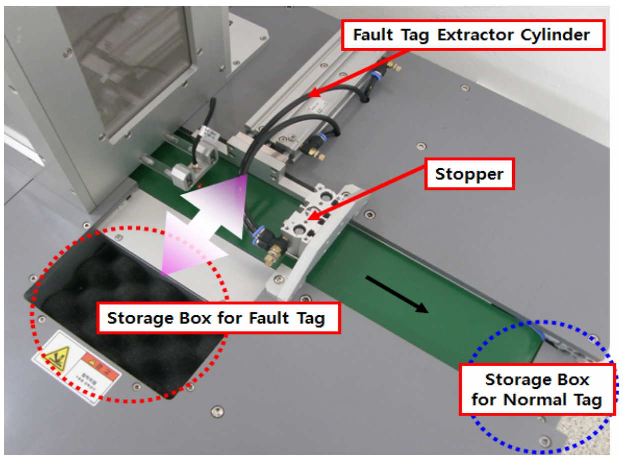
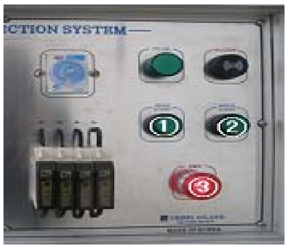

Fault Tag Extractor Cylinder
Stopper
Storage Box for Fault Tag
Storage Box for Normal Tag

Fig.9 Detection facility of defective RFID tags

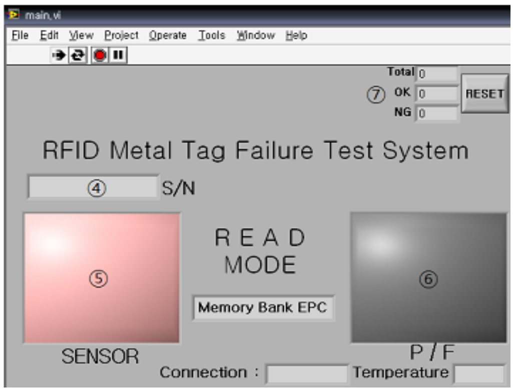
When the sensing result for the tested RFID tag is fault, the '*Stopper*' is operated and the RFID fault tag is moved into the '*Storage Box for Fault Tag*' by the '*Fault Tag Extractor Cylinder*'. The '*Storage Box for Normal Tag*' is used when the tested RFID tag is normal, and the tag is moved along the conveyor.

B. User Interface for Detection for Defective Tags

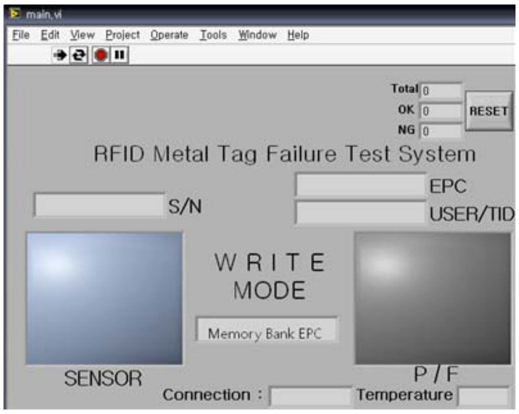
The screen of the program for detection of defective RFID tags is shown in Fig. 10. The test of RFID tag has two modes of READ and WRITE.



(a) Control panel



(b) RFID READ mode



(c) RFID WRITE mode

Fig.10 Screen for detection of RFID defective tags

At first, when the button ① of Fig. 10(a) is pressed, the screen of READ mode shown in Fig. 10(b) is appeared and the test of READ mode is started. When the tag is entered and sensed, the color of area ⑤ is changed into yellow from blue which is the default color. After sensing the tag, the tag ID number(SN; Serial Number) is displayed on the area ④. After the tag is tested, if it is normal(Pass of validation), then the color of area ⑥ is changed into blue, otherwise the color is changed into black(Fail of validation). The numbers of normal/fault tags are displayed on the area ⑦.

The WRITE mode test is started by pressing the button ② of Fig. 10(a), and the WRITE mode screen is appeared as shown in Fig.10(c). The different parts of WRITE mode screen with READ mode are the areas which are displayed for setting the data to be written to RFID tags. The other steps are the same with the READ mode. When the emergent condition happens or the operation stops, the button ③ is pressed.

VI. CONCLUSION

The detection process of the defective tags after the chip bonding job in the manufacturing of RFID tags is accomplished manually, so it has been requested to improve the reliability of the validation checking job of tags. Therefore, we suggest the implementation scheme of the system for detection of the defective tags to make the reliability of the tag production higher by providing the automatic step for the validation checking process of tags instead of the manual job. And, so we provide the basis of the related software to establish the automation system for the detection of the defected RFID tags which is requested in the related industrial field. We use UML modeling technique to design the system which is necessary for the detection of RFID defective tags, and write LabVIEW program using LVOOP which is the object-oriented programming support of LabVIEW in the implementation of the system.

In this paper, we show that it is possible to develop the embedded programs such as the RFID application through the implementation example of the detection system of defective tags by only giving the efforts of UML design level by writing the programs using LVOOP. Also, we could contain the design contents of the program which is developed through the LabVIEW programming process. It is possible to utilize the UML design methodology in the development of the system using LabVIEW, and also possible to make it more easier to maintain the program as well as to develop the system. In the implementation approach of RFID system using LVOOP, the LabVIEW program based on OOP is developed as the object-oriented program and it is possible to apply and associate with the object-oriented development methodology.

By designing the detection system of defective tags according to the UML modeling proposed in this paper, it gives the required functionality and flexibility necessary for the system, so it is possible to apply it to the system of the detection of defective goods in various fields. Also, it is possible to implement the various types of control systems by applying the implementation scheme of LVOOP which is the object-oriented methodology, and its research is continuing.

Petri Net[14] is the methodology which is widely used in the fields of the modeling and simulation of discrete event-driven system. The object-based model using Petri Net can be utilized to prove the development in the implementation approach by the design using UML and implementation using LVOOP as proposed in this paper. In future, we continue the research of utilizing the Petri Net in the design and implementation of LVOOP in the application field of RFID system.

ACKNOWLEDGMENT

This work was supported by Dong-eui University Foundation Grant(2008)

REFERENCES

[1] W. Choi, et al., "The Development of Launch Vehicle Simulator Using an Object-oriented Design," Proc. of Spring Conf. of the Korea Society for Simulation, pp.106-109, 2005.

[2] National Instruments, NI LabVIEW, http://www.ni.com/labview/

[3] D. Beck, et al., "The CS Framework - A LabVIEW Based Approach to SCADA Systems," Proc. of 10th ICALEPCS(Int. Conf. on Accelerator & large Expt. Physics Control Systems, pp.PO1.051-6, 2005.

[4] D. Beck and H. Brand, "Control System Design Using LabVIEW Object Oriented Programming," Proc. of ICALEPCS, pp.84-86, 2007.

[5] M. Chen, "Object Oriented Programming in LabVIEW for Acquisition and Control Systems at the Aerodynamics Laboratory of the National Research Council of Canada," Proc. of 22nd Int. Congress on Instrumentation in Aerospace Simulation Facilities(ICIASF), 2007.

[6] "LabVIEW Object-Oriented Programming: The Decisions Behind the Design," http://zone.ni.com/devzone/cda/tut/p/ id/3574

[7] "LabVIEW Object-Oriented Programming FAQ," http://zone. ni.com/devzone/cda/tut/p/id/3573

[8] "LabVIEW: Why use LVOOP? Answer scalable code," http:// programming.itags.org/labview/41460/

[9] G.Y. Choi, et al., "Trends in RFID Technology and Standardization," Proc. of the Korea Electromagnetic Engineering Society, ETRI, Korea, Vol. 6, No. 5, pp.29-37, 2007.

[10] UML(Unified Modeling Language), http://www.uml.org/

[11] H. Brand, et al., "The PHELIX Control System Based on UML Design Level Programming in LabVIEW," Proc. of ICALEPCS, pp.472-474, 2003.

[12] Vellmen Inc., "K8055/VM110: USB EXPERIMENT INTERFACE BOARD," http://www.funnykit.com/bemarket/ shop/index.php? pageurl=page_goodsdetail&uid=701

[13] ALR-9900 Enterprise RFID Reader Family, "Alien Reader Protocol with ITR Product Overview," http://www. alientechnology.com/docs/product/ DS_ITR.pdf

[14] Petri Nets World, http://www.informatik.uni-hamburg.de/TGI/ PetriNets/

Deok-Gil Jung received the B.S. degree in dept. of computer science from Busan Nat. Univ., Korea in 1983. He received M.S. and Ph.D. degrees in dept. of computer science of Seoul Nat. Univ., Korea in 1986 and 1994, respectively. He has been with dept. of computer science, Dongeui Univ. as a professor from 1986. His research interests are programming languages, especially in 3D/embedded/mobile/robotics programming, and computer education.

Min-Po Jung received the B.S. and M.S. degrees in Computer Science from Ulsan University in 1994 and 1996, respectively. He is currently a professor at Dept. of Cyber Police at YoungSan Univ. since 1999. His areas of research include real time system analysis, LabVIEW software development and UML.

Hyuk-Gyu Cho received the B.S., M.S., and Ph.D. degrees in Computer Science from Pusan Nat. Univ. in 1988, 1990, and 2009, respectively. He is currently a professor at Dept. of Cyber Police at YoungSan Univ. since 2003. His areas of research include digital forensics, ontology, information retrieval, Korean language processing, web application program and object oriented programming language

YoungUhg Lho received the B.S., M.S. and Ph.D. degrees in Dept. of Computer Science from Pusan Nat. Univ., Korea in 1985, 1989, and 1998, respectively. From 1989~1996, he was with the Electronics and Telecommunications Research Institute (ETRI), Daejeon, Korea. Since 1996, he has been with the Dept. of Computer Education, Silla Univ., where he is now Professor. His research interests include intelligent system, ubiquitous computing, embedded system, parallel and distributed system, multimedia system and computer education