UML과 LVOOP를 활용한 RFID 불량 검출 시스템의 구현

정민포* · 조혁규* · 정덕길**

*영산대학교, **동의대학교

The Implementation of the Detection System of RFID Defective Tags Using UML and LabVIEW OOP

Min-Po Jung* · Hyuk-Gyu Cho* · Deok-Gil Jung**

*YoungSan University, ** Dongeui University,

E-mail : minpo@ysu.ac.kr

요 약

RFID 태그 생산 분야에서 RFID 칩 본딩 과정 이후에 RFID 태그 불량 검출 기능을 수행하는 불량 검출 시스템 개발이 요구되어 왔다. 그러나 RFID 태그의 특징을 이해하면서 제대로 된 설계 개념을 가지고 구현된 시스템을 설계하기가 어렵고 사소한 기능의 변화에도 시스템을 처음부터 설계를 해야 하는 어려움이 있었다.

이 논문에서는 RFID 태그 불량 검출 기능을 수행하는 불량검출 시스템을 UML을 이용하여 객체지향 기법으로 설계하고 UML로 설계된 모델링을 객체지향을 지원하는 비주얼 언어인 LabVIEW OOP로 적용하는 방법을 제시한다. UML과 LabVIEW OOP로 설계되고 구현된 불량검출 시스템에 대한 성능과 시스템의 기능 변화에 따른 재설계 기법에 대한 기법도 제안한다.

ABSTRACT

It has been required to develop a defect detection system to perform defect detection capabilities after the bonding process in the production of RFID tags. However, we are difficult to design a system with understanding the characteristics of RFID tags and design concepts. Also we are difficult to modify even minor changes in features.

In this paper, we design the defect RFID detection system using UML and object-oriented design techniques. We suggest the method for apply the UML Diagram to LabVIEW OOP and the technique for redesign the effect detection system's changes.

키워드

UML, LabVIEW OOP, Visual Programming, RFID

Ⅰ. 서 론

RFID 태그 생산 분야에서 RFID 칩 본딩 과정 이후에 RFID 태그 불량 검출 기능을 수행하는 불량 검출 시스템 개발이 요구되어 왔다. 그러나 일반 RFID 관련 소프트웨어 개발 업체에서는 RFID 태그의 특징을 이해하면서 제대로 된 설계 개념을 가지고 구현된 시스템을 설계하기가 어렵고 사소한 기능의 변화에도 시스템을 처음부터 설계를 해야 하는 어려움이 있었다. 이러한 문제점에 대한 주요 이유는 개발 인력이 객체지향의 특징 중인 하나인 재사용성을 전혀 고려하지 않고 개발을 하기 때문이었다.

이에 따라, 이 논문에서는 RFID 태그 불량 검출 기능을 수행하는 불량검출 시스템을 UML을 이용하여 객체지향 기법으로 설계하고 UML로 설계된 모델링을 객체지향을 지원하는 비주얼 언어인 LabVIEW OOP로 적용하는 방법을 제시한다. UML과 LabVIEW OOP로 설계되고 구현된 불량검출 시스템에 대한 성능과 시스템의 기능 변화

에 따른 재설계 기법에 대한 기법도 제안한다.

Ⅱ. 본 론

Ⅱ-1절에서 RFID 태그 불량 검출 프로그래밍을 위한 시스템 모델링을 살펴본다. LabVIEW OOP에서 필요로 하는 UML 다이어그램은 유스케이스 다이어그램, 클래스 다이어그램, 시퀀스 다이어그램이다. Ⅱ-2절에는 만들어진 다이어그램을 LabVIEW OOP 클래스로 생성하는 과정을 보여준다.

Ⅱ-1. RFID 태그 불량 검출 시스템 모델링

실시간 태그 불량 판단 시스템을 설계하기 위해, 표 1에서 설명하고 있는 필수 요구사항을 반영하여 UML을 사용한 최상위 유스케이스(Use Case) 모델[1][2]을 그림 1과 같이 제시한다.

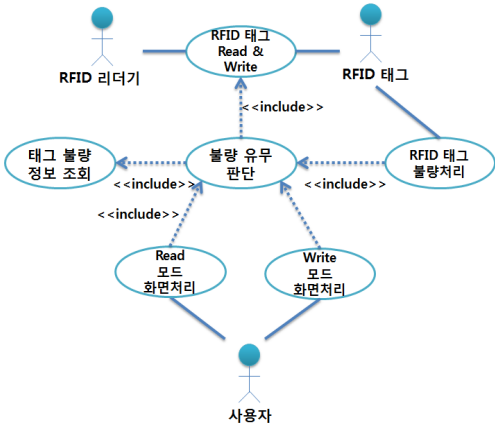

그림 1. 태그 불량 검출 유스케이스 다이어그램

(1) 'RFID 태그 Read & Write' 유스케이스

이 유스케이스는 RFID 태그 메모리에 있는 정보를 Read/Write하는 일반화(generalization) 관계를 유지한다.

(2) '불량 유무 판단' 유스케이스

이 유스케이스는 개발되는 시스템의 핵심 모듈로 태그 정보의 불량 유무에 관한 판단 기능을 수행한다. 또한, '태그 불량 정보 조회', 'RFID 태그 불량 처리', 'Read 모드 화면 처리', 'Write 모드 화면 처리' 유스케이스들과 포함(include) 관계를 유지한다.

(3) '태그 불량정보 조회' 및 '태그 불량 처리' 유스케이스

'태그 불량 정보 조회' 유스케이스는 RFID 태그 정보에 대한 불량 처리 기준을 제공하고, '태그 불량 처리' 유스케이스는 RFID 태그 불량 유무 정보에 따라 실제 하드웨어에서 불량 태그를 처리한다.

(4) 'Read 모드 화면처리'및 'Write 모드 화면처리' 유스케이스

'Read 모드 화면처리' 유스케이스는 태그에 대해 Read 작업을 시도하는 정보에 대한 화면 작업을 처리하며, 'Write 모드 화면처리' 유스케이스는 Write 작업을 시도하는 정보에 대한 화면 작업을 처리한다.

그림 2에서는 불량 태그 검출 시스템에서 사용되는 클래스들을 정의하고 클래스들 간에 존재하는 정적인 관계를 표현할 수 있는 클래스 다이어그램[3]을 보여주고 있다.

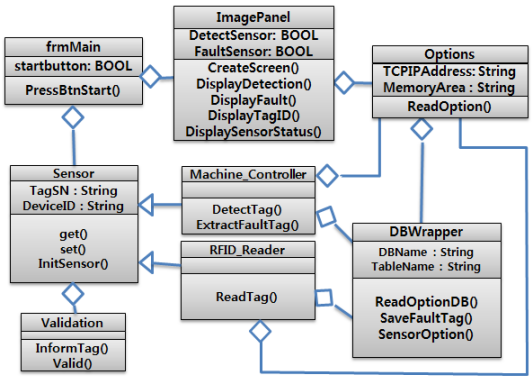

그림 2. 태그 불량 검출 시스템의 클래스 다이어그램

RFID 태그 불량 검출 시스템의 중요 클래스는 frmMain, Sensor, ImagePanel, DBWrapper, Options, Validation으로 분류된다. Sensor 클래스는 Machine_Controller와 RFID_Reader 클래스인 두 개의 서브클래스를 가지고 RFID Reader와 다른 디바이스들 사이의 통신 기능을 담당한다. ImagePanel 클래스는 태그, 센서, 오류 정보 정보를 보여주는 기능을 담당한다. DBWrapper 클래스는 태그 정보와 시스템의 옵션 정보를 저장하는 기능을 담당한다. Options 클래스는 GUI 세팅으로 태그 속성 정보와 통신 정보와 같은 옵션을 설정하는 기능을 담당한다. Validation 클래스는 불량판정을 결정하는 기능을 담당한다.

클래스 다이어그램이 완성되고 나면 UML의 시퀀스(Sequence) 다이어그램을 개발한다[4][5]. 시퀀스 다이어그램은 RFID 불량검출 시스템 내의 객체들 사이의 메시지를 보여준다. 그림 3은 RFID 불량검출 시스템에 대한 시퀀스 다이어그램을 보여준다.

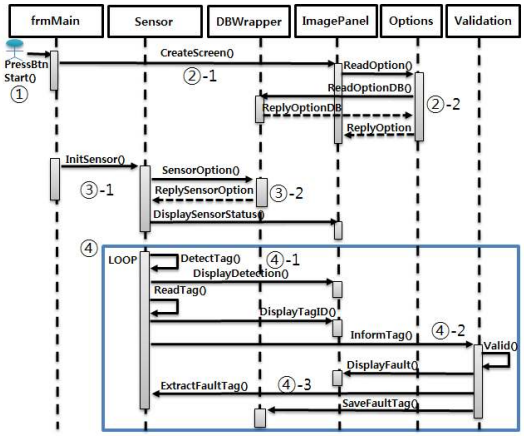

그림 3. RFID 불량 검출 시스템에 대한 시퀀스 다이어그램

① 테스트 시작

RFID 불량 태그에 대한 테스를 시작하기 위한 소프트웨어 버튼이나 하드웨어 장치 버튼을 선택하는 PressBtnStart() 메소드를 호출한다.

② RFID 불량 태그 테스트를 위한 주요 설정하기

②-1. 불량 태그 감지에 대한 화면 설정을 위해 ImagePanel 객체의 CreateScreen() 메소드를 호출한다.

②-2. DBWrapper 객체의 ReadOption 메소드와 Option 객체의 ReadOptionDB() 메소드를 호출하여 데이터베이스로부터 태그 메모리 타입과 정보를 읽는다.

③ 센서 초기화 하기

③-1. frmMain 객체에서 Sensor 객체의 InitSensor() 메소드를 호출하여 센서 초기화를 시도한다.

③-2. DBWrapper 객체의 SensorOption() 메소드를 호출하여 초기값을 읽어 센서 초기화를 시도한다. ImagePanel 객체의 DisplaySensorStatus() 메소드를 호출하여 화면에 값을 출력한다.

④ 센서에서 RFID 태그 정보를 읽은 뒤, 불량 판단하기

④-1. Sensor 객체의 DetectTag() 메소드로 RFID 태그를 감지하고 ImagePanel 객체의 DisplayDetection 메소드로 RFID 태그 감지 정보를 화면에 출력한다. Sensor 객체의 ReadTag() 메소드로 RFID 태그를 읽어들이고 ImagePanel 객체의 DisplayTagID() 메소드로 화면에 태그 ID 값을 출력한다.

④-2. 읽은 태그 정보를 근거로 Validation 객체의 Valid() 메소드를 통해 불량여부를 결정하고 ImagePanel 객체의 DisplayFault() 메소드를 통해 화면에 결정된 불량여부 정보를 출력한다.

④-3. 불량 RFID 태그를 추출하기 위해 Sensor 객체의 ExtractFaultTAG() 메소드를 호출하고 데이터베이스로 불량 정보를 저장하기 위해 DBWrapper 객체의 SaveFaultTag() 메소드를 호출한다.

II-2. LabVIEW OOP 클래스 생성

UML로부터 클래스 다이어그램 명세서가 만들어지면, LabVIEW OOP 클래스[6][7]로 1:1 매핑을 한다. 그림 4는 UML로 설계된 불량 태그 검출 시스템을 예제로 사용하고 있다. UML의 클래스 다이어그램과 클래스 다이어그램을 기초 데이터로 하여 만들어진 LabVIEW OOP에서 생성한 클래스를 보여주고 있다. 그림 4에서 생성된 LabVIEW OOP의 클래스는 UML의 클래스 다이어그램과 정확히 1:1 매핑이 된다.

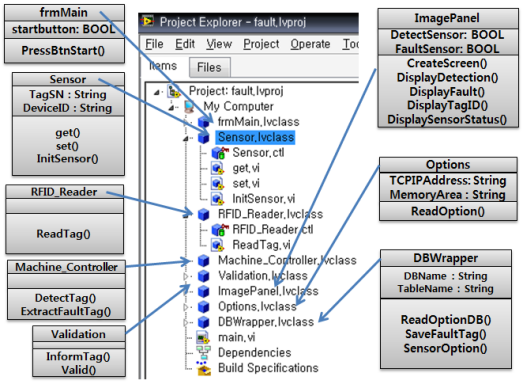

그림 4. 클래스 다이어그램과 LabVIEW OOP 클래스의 관계

생성된 클래스 다이어그램으로부터 LabVIEW OOP의 클래스인 frmMain, Sensor, RFID_Reader, Machine_Controller, Validation, ImagePanel, Options, DBWrapper를 생성한다. Sensor.lvclass는 클래스 이름을 의하고 Sensor.ctl은 클래스 프라이빗 데이터를 의미한다. get.vi, set.vi, InitSenso.vi는 클래스에 포함된 메소드를 의미한다.

표 1. UML 클래스 다이어그램에서 객체트리로의 1:1 변환

UML 클래스 다이어그램	전환	객체 트리
frmMain	→	frmMain
ImagePanel	→	ImagePanel
Sensor	→	Sensor
Machine_Controller	→	Machine_Controller
RFID_Reader	→	RFID_Reader
Validation	→	Validation
Options	→	Options
DBWrapper	→	DBWrapper

표 1에서 UML에서 설계된 각각의 클래스를 LabVIEW OOP로 적용된 결과를 보여준다.

또한, 각각의 UML 클래스의 메소드는 각각의 VI로 변환된다. 표 2에서 UML 클래스의 각 메소드가 객체 트리내의 클래스의 각 VI로 변환되는 것을 정리하였다.

표 2. UML 클래스의 메소드에 대한 LabVIEW OOP의 1:1 매칭 관계

UML 클래스	메소드	전환	VI	객체 트리
frmMain	PressBtnStart()	→	PressBtnStart.vi	frmMain
Sensor	get()	→	get.vi	Sensor
	set()		set.vi	
	InitSensor()		InitSensor.vi	
RFID_Reader	ReadTag()	→	ReadTag.vi	RFID_Reader
Machine_Controller	DetectTag()	→	DetectTag.vi	Machine_Controller
	ExtractFaultTag()		ExtractFaultTag.vi	
ImagePanel	CreateScreen()	→	CreateScreen.vi	ImagePanel
	DisplayDetection()		DisplayDetection.vi	
	DisplayFault()		DisplayFault.vi	
	DisplayTagID()		DisplayTagID.vi	
	DisplaySensorStatus()		DisplaySensorStatus.vi	
Validation	InformTag()	→	InformTag.vi	Validation
	Valid()		Valid.vi	
Options	ReadOption()	→	ReadOption.vi	Options
DBWrapper	ReadOptionDB()	→	ReadOptionDB.vi	DBWrapper
	SaveFaultTag()		SaveFaultTag.vi	
	SensorOption()		SensorOption.vi	

UML로부터 시퀀스 다이어그램 명세서가 만들어지면, LabVIEW OOP 클래스를 이용하여 와이어링(Wiring)을 시작한다. 그림 5는 UML로 설계된 불량 태그 검출 시스템을 예제로 사용하고 있다. UML의 시퀀스 다이어그램과 시퀀스 다이어그램을 기초 데이터로 하여 만들어진 LabVIEW OOP에서 생성한 와이어링을 보여주고 있다. 그림 5에서 생성된 LabVIEW OOP의 와이어링은 UML의 시퀀스 다이어그램과 정확히 1:1 매핑이 된다. ①의 PressStart() 함수는 ImagePanel 객체의 CreateScreen()(②-1) 함수를 호출하게 된다. 이를 LabVIEW OOP에서의 와이어링을 살펴보면, PressBtnStart.vi(메소드)와 ImagePanel.lvclass의 CreateScreen.vi(메소드)와 와이어링이 되어 있는 것을 확인할 수 있다.

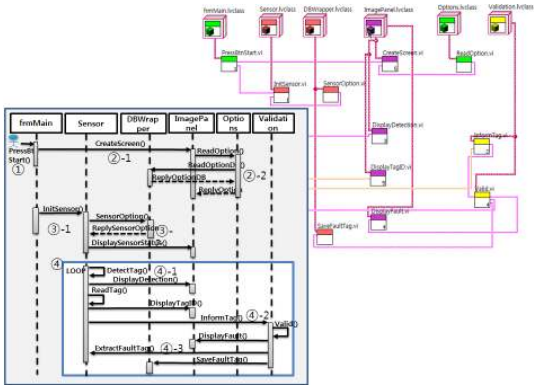

그림 5. 시퀀스 다이어그램과 LabVIEW OOP의 와이어링 관계

III. 결 론

RFID 태그 불량 검출 기능을 수행하는 불량검출 시스템을 UML을 이용하여 객체지향 기법으로 설계하고 UML로 설계된 모델링을 객체지향을 지원하는 비주얼 언어인 LabVIEW OOP로 적용하는 방법을 제시하였다. UML에서 설계된 클래스 다이어그램과 시퀀스 다이어그램은 LabVIEW OOP의 클래스와 블록 다이어그램으로 설계되고 구현된다. 추후, 시스템에 대한 변경으로 인해, UML의 클래스 다이어그램과 시퀀스 다이어그램의 변경된다면 논문에서 제안하는 방법으로 LabVIEW OOP 프로그램을 변경할 수 있다. 이는 LabVIEW OOP에 대한 설계 개념을 포함하는 재사용성을 높여 LabVIEW OOP 개발자들에게 객체지향 특징인 재사용성을 높여 프로그램 개발에 도움을 준다.

참고문헌

[1] D.Rosenberg, K. Scott, Use case driven object modeling with UML: a practical approach, Addison-Wesley, 1999.

[2] D. Leffingwell, D. Widrig, “Managing Software Requirements: A Use Case Approach”, Pearson Education, 2003.

[3] T. Levendovszky, L. Lengyel, “A UML class diagram-based pattern language for model transformation systems”, World Scientific and Engineering Academy and Society (WSEAS), 2005.

[4] T. K, Extracting sequence diagram from execution trace of Java program, Principles of software Evolution, 2005.

[5] D. Bell, “UML's Sequnce Diagram”, IBM, 2004.
[6] National Instrument, "LabVIEW 8.5," http://www.ni.com/labview/
[7] W.-K. Jehng, "Using LabView to Integrate RFID system and Database for Supply Chain Efficiency Improvement", International Journal of Intelligent Control and Systems, vol. 13, No. 3, pp.189-195, 2008.