RFID 리더 투명성 지원을 위한 리더 프레임워크 개발

백선재* · 문미경**

Development of an Reader Framework for Transparency in RFID Reader

Sunjae Baek* · Mikyeong Moon**

요 약

최근 RFID (Radio Frequency Identification) 기술이 다양한 분야에서 관심을 받고 사용됨으로써 RFID 리더 제조회사들은 다양한 종류의 리더를 생산하고 있다. 현재 RFID 기술에서 RFID 태그 데이터 전송 형태는 EPCglobal에서 제공하는 표준 규약을 따르고 있지만 각 리더 별 리더 연결 및 통신 방법은 회사별로 다른 실정이다. RFID 리더를 사용하던 중 타 회사나 리더 인터페이스가 다른 RFID 리더로 변경하거나 새롭게 추가해야 하는 경우에는 사용자가 직접 RFID 미들웨어 개발단계로 돌아가 미들웨어의 내부 요소를 변경해야 하는 상황이 발생한다. 본 논문에서는 RFID 기술을 사용 시, 특히 여러 회사 또는 표준화되지 않은 인터페이스를 제공하는 RFID 리더를 사용하는 경우에 RFID 애플리케이션과 독립적으로 리더를 관리할 수 있는 리더 프레임워크를 개발하고자 한다. 본 리더 프레임워크는 RFID 애플리케이션과 RFID 리더 사이에 존재하여 RFID 리더 투명성을 제공한다. 본 리더 프레임워크는 현재 작동되고 있는 리더를 모니터링 할 수 있고, 리더의 속성을 확인할 수 있으며 RFID 태그 이벤트 정보, 시스템 로그 정보를 확인할 수 있다. 본 리더 프레임워크를 통해 수십 대의 이기종의 RFID 리더 상태를 용이하게 관리할 수 있게 되며 애플리케이션 상의 코드 수정 없이 리더의 추가, 삭제가 가능하게 된다.

ABSTRACT

More recently, variety RFID (Radio Frequency Identification) readers are produced by RFID equipment manufactures. Although a transmission standard instituted by EPCglobal is proposed for data transmission between the RFID readers and tags, other RFID reader protocols and the communication connection methods are be in use in other RFID companies. To replace or add the RFID readers of an RFID system, the developers should make changes to the core of the application and/or middleware. In this paper, we propose an RFID reader framework which can manage RFID readers without having to make changes the code of the application in environment with the growing number of heterogeneous RFID readers.This framework that sits on the layer between the RFID readers and the applications provides transparency to the RFID readers. Additionally, it can be used for monitoring the state and the property of all connected RFID, and for recording the RFID tag event logs and system logs.By using this framework, heterogeneous readers can be replaced and added without writing additional code in the applications. Consequently the readers can be easily managed and controlled by the RFID system administrator.

키워드

RFID, RFID 리더, 리더 프레임워크, 투명성

Key word

RFID, RFID Reader, Reader Framework, Transparency

* 준회원: 동서대학교
** 정회원: 동서대학교 (교신저자, mkmoon@dongseo.ac.kr)

접수일자 : 2010. 09. 18
심사완료일자 : 2010. 11. 26

Ⅰ. 서 론

RFID (Radio Frequency Identification) 미들웨어는 RFID 리더와 외부 시스템 (Enterprise Resource Planning: ERP, Supply Chain Management: SCM, Customer Relationship Management: CRM 등)의 중간에 위치하여 다양한 RFID 시스템 환경에서 생성되는 대량의 정보를 수집하여 이를 분석한 후 의미 있는 정보만을 응용 시스템에 전달하는 소프트웨어이다[1]. RFID 기술을 도입한 예를 살펴보면 각 시스템들은 매장 내 스마트선반, 매장 출입구, 물류센터, 항만 입구, 컨테이너 블록, 크레인 등 여러 장소 또는 시설물에 RFID 리더를 부착하여 상품 또는 컨테이너의 정보를 습득한다[2]. 이러한 RFID 리더들은 모두 같은 회사, 같은 종류의 제품을 사용할 수 있다. 그러나 RFID 리더를 부착하는 위치 또는 장소에 따라서 RFID 리더의 크기와 종류가 달라질 수도 있고 그 크기에 따라 여러 회사의 RFID 리더를 사용할 수 있다.

이때 이기종의 RFID 리더마다 제공하는 인터페이스가 서로 상이하여 모든 RFID 리더로부터 태그 값을 전송 받기 위해서는 각 리더가 제공하는 인터페이스를 모두 정의해야 한다. 또한 특정 장소에 있는 RFID 리더에 알 수 없는 오류가 발생하거나 파손 또는 고장이 발생하여 부득이하게 교체해야 할 때 동일 회사의 제품이 없어 다른 회사의 제품으로 사용해야 할 경우에 제공되는 인터페이스를 정의하기 위하여 RFID 미들웨어를 재구성해야 한다. 이렇게 이기종 RFID 리더를 사용하는 환경에서 RFID 리더가 교체되거나 새로운 RFID 리더를 추가하는 경우 RFID 미들웨어 또는 상위 애플리케이션을 다시 수정해야 하는 경우가 발생한다[3]. 이에 따라 RFID 표준단체인 EPCglobal에서는 ALE (Application Level Events)라는 RFID 미들웨어를 제시하였다[4]. 그러나 표준을 따르는 RFID 미들웨어는 그 자체가 복잡하고 부피가 크며 가격이 높기 때문에 이를 기반으로 한 시스템은 개발 속도가 늦어지고, 유지보수가 힘들어진다.

본 논문에서는 RFID 애플리케이션과 독립적으로 리더를 관리할 수 있는 리더 프레임워크를 제안한다. 본 프레임워크의 주요 기능은 많은 리더가 부착된 RFID 시스템 운용 시, 리더의 추가, 삭제, 대체와 같은 변경이 발생했을 때, 애플리케이션의 변경 없이 유연하게 변경처리 될 수 있도록 하는 것과 이러한 변경을 관리자가 바로 인지할 수 있도록 모니터링하고 알려줄 수 있도록 하는 것이다. 이를 위해 먼저, 리더 관리 시 필요한 공통적 기능들과 이와 함께 다루어져야 하는 각종 속성들을 분석한다. 본 논문에서는 인터페이스가 일치하지 않는 클래스를 함께 운용해야 하는 경우에 사용할 수 있는 어댑터 패턴 (adapter pattern)을 적용하여 리더를 관리하기 위한 기능을 추상화한 추상 리더 클래스를 제공한다. 또한 각각의 리더마다 가지고 있는 리더 속성들과 리더 관리 시 변경이 잦은 연동정보들을 클래스에서 분리시키고 이를 리더 구성 XML 파일로 만들어 이용하도록 한다. 또한 리더 프레임워크 매니저를 제공하여 현재 리더 프레임워크와 연동되어 있는 RFID 리더의 정보를 그래픽 인터페이스를 통해 실시간으로 모니터링 할 수 있고 상태 변경에 대해서 알림을 줄 수 있는 기능을 가지도록 한다. 본 리더 프레임워크는 리더와 애플리케이션 간에 의사소통을 할 수 있는 중개자 역할을 하며, 원거리 통신을 위해 무선 인터페이스 프로토콜을 사용하여 다양한 애플리케이션을 지원한다.

본 논문의 구성은 다음과 같다. 2장에서는 RFID 미들웨어에 대한 설명과 이기종의 RFID 리더 연동방법에 대한 관련연구를 기술한다. 3장에서는 RFID 리더 투명성 지원을 위한 리더 프레임워크를 설명하고, 4장은 리더 프레임워크 개발 결과를 보여준다. 마지막으로 5장에는 결론을 제시한다.

Ⅱ. 관련 연구

2.1 RFID 미들웨어

RFID 시스템은 그림 1과 같이 태그, 리더, 미들웨어 및 애플리케이션으로 구성되고, 유무선 통신망과 연동되어 사용된다. 태그는 객체를 인식할 수 있는 정보를 가지고 객체 상에 위치한다. 리더는 객체의 정보를 수집 처리하며, 송신 및 수신 기능을 가진다.

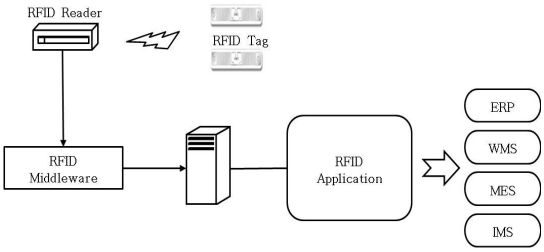

그림 1 RFID 시스템 구성도
Fig. 1 RFID system architecture

RFID 미들웨어는 리더에서 계속적으로 발생하는 식별코드 데이터를 수집, 제어, 관리하는 기능을 하며, 모든 구성요소와 연결되어 계층적으로 조직화되고 분산된 구조의 미들웨어 네트워크를 구성하여 서로 통신한다. RFID 표준단체인 EPCglobal에서 제시한 ALE는 EPC (Electronic Product Code) 처리 시스템에서 RFID 태그를 인식한 리더가 애플리케이션 계층으로 데이터를 전달하는 역할을 하는 일종의 미들웨어이다[5]. 여기서는 미들웨어와 리더 사이의 프로토콜을 정의하고 있는 EPCglobal Reader Protocol v1.1과 이를 관리하는 EPCglobal Reader Management Protocol v1.0.1을 제시하고 있다. 그러나 현재 출시된 여러 업체들의 RFID 리더들은 이러한 표준들을 따르지 않고 독자적인 프로토콜을 사용하는 경우가 많다.

2.2 이기종의 RFID 리더 연동 방법

현재 EPCglobal에서 제안하는 표준 프로토콜과 독자적인 프로토콜을 사용하는 모든 RFID 리더 연동을 위해 RFID 미들웨어와 다양한 리더 프로토콜을 통합적으로 관리하기 위한 연구가 활발히 진행되고 있다. [6]의 연구에서는 비표준 리더가 사용하는 프로토콜을z 표준 리더가 제공하는 프로토콜로 변경해 주는 CustomAdapter를 가진 RPManager를 제시하였다.

CustomAdapter를 통해 다수의 비표준 리더를 연결하여 표준 리더처럼 사용할 수 있는 기능을 제공한다. UNIAplusRFID라는 제품은 RFID 리더에 대한 추상화로 plug-in방식을 선택하고 있다. 이를 통해 새로운 장치를 추가할 때 RFID 리더의 특성을 구현한 plug-in과 속성 meta data를 추가하여 시스템 확장을 가능하게 한다[7]. 또한 온톨로지 기반으로 다양한 RFID 리더를 처리하는 방법을 제안하는 연구가 있다[8]. 이 연구에서는 RFID 리더들의 미들웨어 접속 정보 및 데이터 프로토콜 정보를 온톨로지 메타 데이터로 구축하여, 다양한 RFID 리더를 RFID 미들웨어에서 지원할 수 있도록 한다. 이처럼 효율적으로 이기종의 RFID 리더를 연동시키기 위해 다양한 방법들이 제시되고 있다. 그러나 이러한 방법들을 지원하기 위한 도구 및 리더 관리 환경을 통합적으로 제시하지 않고 있다. 본 논문에서는 프레임워크를 통해 RFID 리더를 추가, 삭제 및 대체할 수 있는 방법을 지원할 수 있도록 할 뿐만 아니라 RFID 리더의 현재 상태, 인식되는 태그 정보 등을 관리할 수 있도록 하는 통합적인 환경을 제시한다.

Ⅲ. RFID 리더 투명성 지원을 위한 리더 프레임워크

3.1 RFID 리더 프레임워크 요구사항

본 논문에서는 제안하는 RFID 리더 프레임워크에 대한 요구사항은 표 1과 같다.

표 1. 리더 프레임워크 요구사항
Table. 1 Requirements of Reader Framework

요구사항 명	요구사항 설명
리더 연동 관리	이기종의 RFID 리더 연동(추가, 삭제, 교체)이 가능하다. 새로운 리더가 연동되거나 다른 리더로 교체 되는 경우 애플리케이션이 수정되지 않도록 한다.
리더 상태 관리	연동되어 있는 리더의 상태를 조회하고 상태 변경에 대한 알림을 받는다.
리더 정보 관리	리더의 정보를 관리한다. 리더가 태그를 인식하는 간격이나 주기를 관리한다. 리더와 연결되어 있는 안테나의 정보를 관리한다.
로그 관리	태그를 인식한 리더의 정보와 인식된 태그의 정보를 관리한다. 시스템 내부 동작현황을 관리한다.
리더 모니터링	리더의 구성과 정보, 상태를 모니터링한다.

3.2 RFID 리더 프레임워크 아키텍처

그림 2는 RFID 리더 프레임워크에 대한 아키텍처이다. 리더 프레임워크는 *Reader Manager, RFID Tag Event Manager, Log Manager, Network Manager, Client Request Manager, Query Manager*로 구성된다.

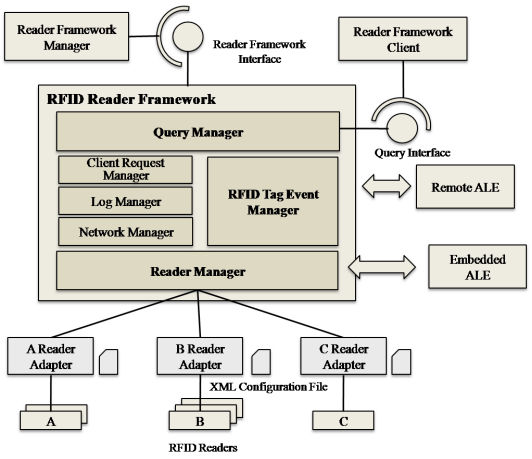

그림 2. RFID 리더 프레임워크의 아키텍처
Fig. 2 Architecture of RFID reader framework

*Reader Manager*는 이기종의 RFID 리더와의 연동을 위해 추상 리더 클래스를 상속받은 복수개의 *Reader Adapter*를 관리한다. 또한 리더 구성 XML 파일 내의 정보에 따라 *Reader Adapter*와 RFID 리더를 관리한다.

*RFID Tag Event Manager*는 전달 받은 RFID 이벤트를 분석한다. 분석한 RFID 이벤트를 통해 인식된 태그 또는 RFID 리더의 상태를 수정하고 필요한 경우에는 다른 RFID 이벤트로 재구성하여 다른 매니저들에게 전달한다. *Query Manager*는 RFID 애플리케이션으로부터 *Query Interface*를 통해 RFSpec (Reader Framework Specification)을 받아 질의 조건을 등록한다. 질의 완료된 검색 결과는 RFReport (Reader Framework Report)를 통해 RFID 애플리케이션에게 원격 통신 (.Net Remoting)을 통해 송신한다. *Reader Framework Interface*는 리더 프레임워크를 관리하고 모니터링 할 수 있는 유저 인터페이스이다. 이를 통해 Reader Adapter의 구성, RFID 리더 속성 Log, Remote Client, Query를 확인할 수 있다.

3.3 RFID 리더 관리 방법

- 리더 연동기능 정의

리더 프레임워크는 RFID 리더의 추가, 삭제, 변경을 위해 리더의 공통적 핵심기능을 관리한다. 리더의 핵심기능에는 리더 초기화, 리더 연결, 리더 연결 해제, 리더 종료, 태그 인식, 태그 인식 종류, 리더 상태 획득 등이 있다. 리더의 이러한 핵심기능을 하나의 공통된 요소로 관리하기 위해 어댑터 패턴을 사용하여 이를 추상 리더로 표현한다. 어댑터 패턴을 이용하여 애플리케이션과 리더 사이의 연관성을 제거 할 수 있고 차후에 리더가 추가, 삭제, 변경되어도 애플리케이션은 변경되지 않게 할 수 있다. 다음 그림 3은 어댑터 패턴을 적용한 추상 리더의 클래스 다이어그램이다.

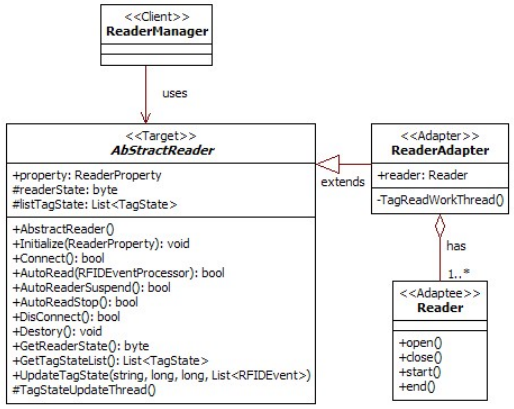

그림 3. 추상 리더의 클래스 다이어그램
Fig. 3 Class diagram of an abstract reader

- 리더 연동정보 정의

리더 프레임워크는 RFID 리더 연동을 위해 리더 자체의 정보, 태그인식 설정 정보, 안테나 정보를 관리해야 한다. 리더정보에는 리더 타입, 리더 고유명, 리더 논리적 이름, ip주소 등이 있다. 태그인식 설정정보에는 태그 인식주기, 태그 상태 변경주기 등이 있다. 안테나 정보에는 리더에 포함된 안테나의 이름, 위치, 수신범위 등이 있다. 리더 연동을 위하여 필요한 이러한 정보들은 리더가 변경될 때마다 수정되어야 하는 정보이기 때문에 이를 클래스와 분리하여 단독의 리더 구성 XML 파일 (Reader Configuration XML File)로 관리한다. 이를 통해 새로운 리더를 연결하거나 삭제, 대체할 경우에 리더 구

성 XML 파일만 수정하면 되게 한다. 즉, RFID 리더와의 연동을 위해 리더 구성 XML 파일만 정의, 재수정 하게 되므로 애플리케이션은 리더의 변동에 영향을 받지 않는다. 그림 4는 리더 연동정보를 기술할 수 있는 리더 구성 XML 스키마 파일 내용이다.

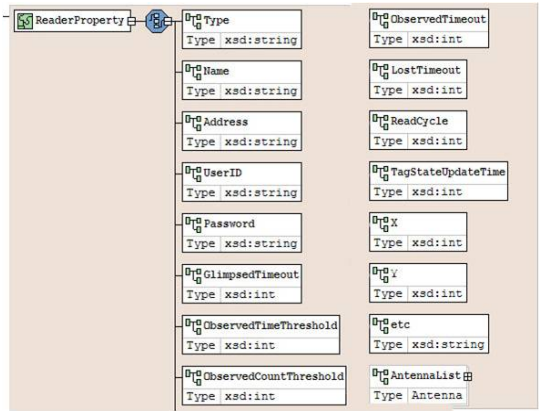

그림 4. 리더 구성 XML 스키마
Fig. 4 Reader configuration XML schema

• 리더 상태관리기능 정의

리더 프레임워크는 동작하는 RFID 리더의 상태를 파악하기 위해 리더의 상태를 실시간으로 조회할 수 있어야 한다. 리더의 상태는 리더가 접속이 되지 않은 상태 (Disconnected), 리더 접속이 완료된 상태이나 읽기 명령을 수행하지 않은 상태 (Connected), 리더가 읽기 명령을 수행하고 있는 상태 (ReadingStarted), 리더가 읽기를 잠시 중지한 상태 (ReadingSuspended), 그리고 하드웨어나 소프트웨어적인 문제로 리더 동작이 정상적이지 않은 상태 (Error)로 분류한다. 리더 프레임워크는 리더의 이러한 상태를 조회하는 단순한 모니터링 기능뿐만 아니라 리더의 상태가 변경되는 경우 리더의 상태가 변경되었다는 알림을 받을 수 있게 한다. 리더 상태 알림은 리더가 추가 되었을 경우 (AddReader), 리더가 삭제되었을 경우 (RemoveReader), 리더 상태가 변경 되었을 경우 (ReaderStateChange)로 구분하여 발생시킨다.

3.4 RFID 이벤트 및 메시지 처리 방법

현재 존재하는 RFID 미들웨어의 경우 다양한 형태의 RFID 리더를 지원함에 있어 ISO와 EPCglobal에서 제안한 표준을 바탕으로 한 장비들이 주를 이루고 있다 [8]. 본 리더 프레임워크는 EPCglobal Reader Protocol Standard v1.1 (RP)을 바탕으로 구성된다. RP에서는 RFID 태그 데이터의 중복 제거를 위해 이벤트 개념을 사용하며, 태그가 리더에 읽히게 되면 각 태그는 RPState를 가지게 된다. 본 리더 프레임워크에서 사용한 RPState의 정의와 RPEvent의 정의는 다음 표 2, 표 3과 같다.

표 2. RPState의 정의
Table. 2 Definition of RPState

이름	설명
isUnknown	태그의 상태를 알 수 없는 상태이다. 이는 태그가 리더 인식영역을 완전히 벗어났다는 것을 의미한다.
isGlimpsed	태그가 간헐적으로 읽히고 있는 상태이다. 이는 태그가 불안정적으로 인식되고 있다는 것을 의미한다.
isObserved	태그가 안정적으로 읽히고 있는 상태이다.
isLost	인식되고 있던 태그가 인식되지 않는 상태이다. 이는 태그가 리더 인식 영역을 벗어났다는 것을 의미한다.

표 3. RPEvent의 정의
Table. 3 Definition of RPEvent

이름	설명
evNew	태그가 최초로 인식되었을 경우에 발생한다.
evGlimpsed	인식 되지 않던 태그가 인식 될 경우에 발생한다.
evObserved	태그가 안정적으로 인식 되는 상황이다. 태그의 인식은 사용자가 지정한 특정시간 이상 읽힐 경우에 발생한다.
evLost	안정적으로 인식되던 태그가 특정시간 이상 인식되지 않을 경우에 발생한다.
evPurged	태그가 인식되지 않는 시간이 오랜 시간이 지났을 경우에 발생한다.

RPState는 태그의 인식 시간에 따라 주기적으로 변화하게 되며 상태가 변경되는 경우 RPEvent를 발생시킨다. 리더는 인식 범위 내에 있는 모든 태그들의 RPState를 관리하게 되는데 예를 들어 리더에 10개의 태그가

인식되고 있을 경우 리더는 10개의 태그 RPState를 관리한다.

그림 5는 리더 프레임워크 내에서 발생되는 RFID 이벤트의 흐름을 도식화 한 것이다.

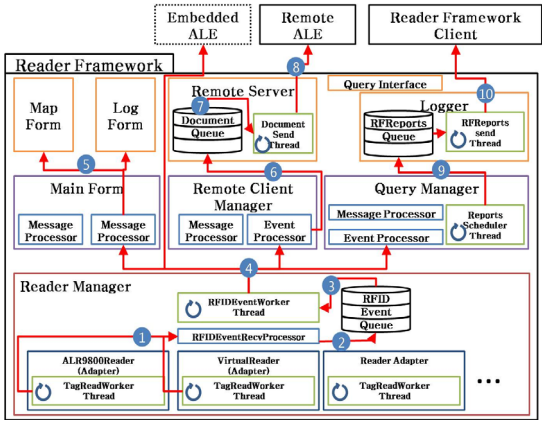

그림 5. RFID 이벤트 흐름
Fig. 5 RFID event flow

① 각 어댑터에서 RFID 이벤트가 생성되면 리더 매니저의 RFIDEventRecvProcessor에게 전달한다. 리더 프레임워크의 성능향상을 위해서 각 어댑터에서는 RawData를 기반하여 RFID 이벤트를 생성하고 이때 스무딩 필터링 (smoothing filtering)을 적용하여 데이터의 중복을 제거한다. RFID 이벤트의 생성은 추상 리더 어댑터에 정의되어 있으며 새로운 어댑터를 추가하는 경우 인식된 태그 ID 및 태그인식 설정 정보만 업데이트 해주면 된다.

② 수신된 RFID 이벤트는 리더 매니저에 포함된 저장소 (RFIDEventQueue)에 저장한다. 리더 어댑터로부터 받은 RFID 이벤트를 바로 처리하는 경우 리더 매니저와 다른 매니저들 사이에 오류가 발생할 수 있다. 이로 인해 RFID 이벤트를 처리하는 과정이 지연되어 태그인식이 늦어지거나 인식하지 못하는 경우가 발생할 수 있다. 이를 제거하기 위해 각 리더 어댑터로부터 수신된 RFID 이벤트를 바로 처리 하지 않고 저장소에 저장한다.

③ 각 매니저에게 RFID 이벤트를 전달하기 위해서 저장소에 저장되어 있는 RFID 이벤트가 존재하는지 확인한다.

④ RFID 이벤트를 요청한 매니저에게 RFID 이벤트를 전달한다. 주기적으로 저장소에 저장되어 있는 RFID 이벤트가 있는지 확인한 후에 저장되어 있는 RFID 이벤트가 존재하는 경우 각 매니저에게 RFID 이벤트를 전달한 후에 저장소에 저장되어 있는 RFID 이벤트를 삭제한다. RFID 이벤트를 요청하는 매니저에는 Main Form, Embedded ALE, Remote Client Manager, Query Manager가 있다.

⑤ Main Form에서는 수신된 RFID 이벤트를 Map Form 및 Log Form에 전달하여 배경화면, 로그화면에 출력된다.

⑥ Remote Client Manager는 수신된 RFID 이벤트를 다시 전송하기 위해 Remote Server에 있는 Document Queue에 저장한다.

⑦ Remote Server는 Remote ALE에게 RFID 이벤트를 전달하기 위해서 Document Queue에 저장되어 있는 RFID 이벤트가 존재하는지 확인한다.

⑧ Remote Server는 현재 접속되어 있는 모든 클라이언트에게 RFID 이벤트를 문서화 시킨 EventDocument를 전송한다.

⑨ Query Manager는 현재 전송해야 하는 쿼리 및 수신 클라이언트가 있을 경우 Logger에 RFReports (Reader Framework Reports)를 전달한다. Query Manager에 전달된 RFID 이벤트는 RPEvent 상태로 관리된다.

⑩ Logger는 Reader Framework Client에게 RFReports를 전달한다.

Ⅳ. 리더 프레임워크 개발 결과

본 논문에는 RFID 리더 프레임워크를 이용하여 실제 RFID 리더를 연동해 보았다.

- ALR-9800 리더 연동

실제 RFID 리더를 연동한 사례는 Alien사의 ALR-9800 모델이다. 먼저 ALR-9800 연동을 위해서 ALR9800 Reader 어댑터를 생성한 후에 Alien사에서 제공하는 라이브러리 파일을 추가하여 리더 연결 및 통신을 수행하게 한다. 다음에는 리더 프레임워크에 있는 Device Property에 새로운 리더의 정보 즉, ALR-9800의 위치와 태그인식 설정 정보를 작성한다. 다음 그림 6은 실제 생

성된 ALR9800Reader 어댑터의 클래스 다이어그램이고, DeviceProperty의 XML 파일의 내용은 다음 그림 7과 같다.

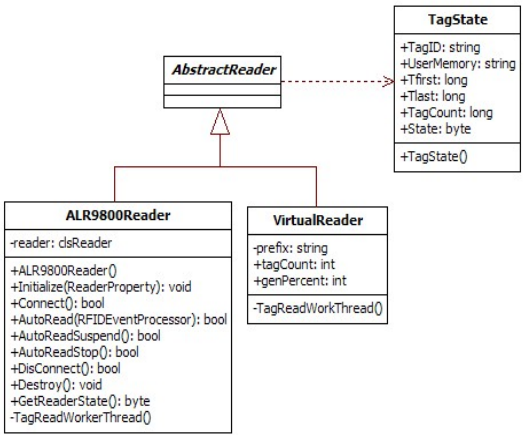

그림 6. ALR9800Reader 클래스 다이어그램
Fig. 6 Class diagram of an ALR9800 reader

```
<?xml version="1.0" encoding="utf-8"?>
<DeviceProperty xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
  <ReaderPropList>
    <Type>AlienALR9800</Type>
    <Name>Door1</Name>
    <Address>192.168.1.100:23</Address>
    <UserID>alien</UserID>
    <Password>password</Password>
    <GlimpsedTimeout>400</GlimpsedTimeout>
    <ObservedTimeThreshold>400</ObservedTimeThreshold>
    <ObservedCountThreshold>0</ObservedCountThreshold>
    <ObservedTimeout>400</ObservedTimeout>
    <LostTimeout>400</LostTimeout>
    <ReadCycle>200</ReadCycle>
    <TagStateUpdateTime>100</TagStateUpdateTime>
    <X>100</X>
    <Y>100</Y>
    <AntennaList>
      <Name>ANT1</Name>
      <X>70</X>
      <Y>110</Y>
      <Angle>110</Angle>
      <DetectionnAngle>60</DetectionnAngle>
      <DetectionnRange>70</DetectionnRange>
    </AntennaList>
    <AntennaList>
      <Name>ANT2</Name>
      <X>130</X>
      <Y>110</Y>
      <Angle>10</Angle>
      <DetectionnAngle>60</DetectionnAngle>
      <DetectionnRange>70</DetectionnRange>
    </AntennaList>
  </ReaderPropList>
</DeviceProperty>
```

그림 7. ALR-9800을 추가한 DeviceProperty.xml 파일
Fig. 7 DeviceProperty.xml file added ALR-9800

- 기존 리더의 변경, 추가, 삭제

기존의 리더를 변경, 추가, 삭제하는 경우에는 리더 어댑터를 추가할 필요 없이 리더 구성 XML 파일을 재구성 하면 된다.

다음 그림 8은 실제 ALR9800Reader를 연동한 후 추가적으로 리더를 연동 했을 때의 DeviceProperty의 XML 파일의 내용이다. 그림에서 사각형 표시부분이 추가되는 내용이다.

```
<?xml version="1.0" encoding="utf-8"?>
<DeviceProperty xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
  <ReaderPropList>
    <Type>AlienALR9800</Type>
    <Name>Door1</Name>
    <Address>192.168.1.100:23</Address>
    <UserID>alien</UserID>
    <Password>password</Password>
    <GlimpsedTimeout>400</GlimpsedTimeout>
    <ObservedTimeThreshold>400</ObservedTimeThreshold>
    <ObservedCountThreshold>0</ObservedCountThreshold>
    <ObservedTimeout>400</ObservedTimeout>
    <LostTimeout>400</LostTimeout>
    <ReadCycle>200</ReadCycle>
    <TagStateUpdateTime>100</TagStateUpdateTime>
    <X>100</X>
    <Y>100</Y>
    <AntennaList>...</AntennaList>
    <AntennaList>...</AntennaList>
  </ReaderPropList>
  <ReaderPropList>
    <Type>AlienALR9800</Type>
    <Name>Door2</Name>
    <Address>192.168.1.100:24</Address>
    <UserID>alien</UserID>
    <Password>password</Password>
    <GlimpsedTimeout>400</GlimpsedTimeout>
    <ObservedTimeThreshold>400</ObservedTimeThreshold>
    <ObservedCountThreshold>0</ObservedCountThreshold>
    <ObservedTimeout>400</ObservedTimeout>
    <LostTimeout>400</LostTimeout>
    <ReadCycle>200</ReadCycle>
    <TagStateUpdateTime>100</TagStateUpdateTime>
    <X>100</X>
    <Y>100</Y>
    <AntennaList>...</AntennaList>
    <AntennaList>...</AntennaList>
  </ReaderPropList>
</DeviceProperty>
```

그림 8. 변경된 DeviceProperty.xml 파일
Fig. 8 revision DeviceProperty.xml file

- 리더 상태 조회 화면

그림 9는 본 논문을 통해 개발된 RFID 리더 프레임워크의 사용자화면이다. 본 화면의 우측상단에는 지도화면, 하단에는 로그화면, 좌측상단에는 장치그룹, 하단에는 리더의 정보가 나타나 있다. 지도화면에는 현재 가상의 냉동창고 도면이 나타나 있으며, 2개의 가상리더가 배치되어 있다. 그리고 리더의 양 옆에는 각 리더의 안테나의 인식범위가 표시되어 있다. 태그의 수신 상태가 양호 할 경우에는 초록색 신호가 점등되며 인식된 태그 ID가 화면에 출력된다. 로그화면에서는 태그 이벤트 로그와 시스템 로그를 확인할 수 있다. 장치그룹에서는 현재 추가되어 있는 리더 어댑터를 확인할 수 있고 각 어댑터별로 연결되어 있는 리더들을 확인할 수 있다. 리더 정보 화면에서는 리더 연동 시에 작성한 리더 구성 XML 파일의 내용이 나타나있다.

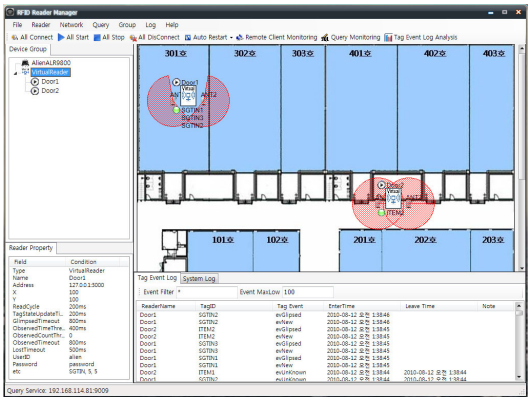

그림 9. RFID 리더 프레임워크 결과 화면
Fig. 9 GUI of RFID reader framework

• 클라이언트 화면

그림 10은 Query Interface를 참조해 만들어진 클라이언트화면이다. 클라이언트는 5초 동안 5개의 태그 값을 획득한다는 query를 한 후 그에 따라 전송 받은 데이터를 화면에 출력한 것이다.

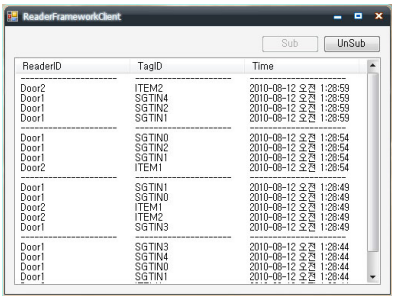

그림 10. 리더 프레임워크 클라이언트 화면
Fig. 10 Client GUI of reader framework

V. 결 론

본 논문에서는 RFID 애플리케이션 사용 시 RFID 리더의 연동에 있어 애플리케이션과 리더 사이의 연관성을 배제하여 리더 투명성을 지원하는 리더 프레임워크에 대해 연구하였다. 리더 프레임워크는 RFID 애플리케이션과 RFID 리더의 연관성을 줄이기 위해 리더 어댑터를 제공한다. 리더 어댑터를 통하여 RFID 리더 연동을 쉽게 할 수 있으며, 리더 프레임워크는 리더 어댑터를 관리해줌으로써 리더의 상태를 빠르게 파악할 수 있다. 또한 현재 동작하고 있는 RFID 리더의 상태를 모니터링 할 수 있는 관리자화면이 제공됨으로써 RFID 리더의 동작이 멈춘 경우 즉각적으로 인지하여 조치를 할 수 있게 해준다.

본 리더 프레임워크를 사용함으로써 RFID 애플리케이션의 유지보수성을 높여주며, RFID 시스템 구성요소 각각의 물리적 인식률뿐만 아니라 전체 시스템의 동작과 관련되는 논리적 인식률을 높여주는 역할을 한다.

참고문헌

[1] 유승화, 유비쿼터스 사회의 RFID, 전자신문사, 2005.

[2] 대한무역투자진흥공사 (http://www.kotra.or.kr), 시장개척사업, 시장조사

[3] E. Aboulouz, D. Deugo, "RFIDMania Extensible and adaptable RFID Middleware and Specifications", IEEE Cybertics and Intelligent Systems, pp.355-361, 2010.

[4] EPCglobal, "The Application Level Events (ALE) Specification, Version 1.1 Part 1: Core Specification", EPC global Last Call Working Draft, May 2007.

[5] MetaBiz (http://meta-biz.net), ALE System Architecture

[6] 안종민, 송하주, "이기종 리더의 통합접속 및 관리를 위한 RPManager의 설계와 구현", 한국정보과학회, Vol.35, No.2, pp.92~97. 2008.

[7] 한미IT, UNIA plus RFID, http://www.hanmiit.co.kr/

[8] 노영식, 변영철, 이동철, "ALE 미들웨어를 위한 다양한 RFID 리더 처리 방법", 한국콘텐츠학회논문지, Vol.9, No.4, pp.55~64, 2009.

[9] EPCglobal, "Reader Protocol Standard, Version 1.1", EPCglobal, Jun 2006.

[10] M.K. Moon, K. Yeom, "Product Line Architecture for RFID-Enabled Applications", In Proceeding of 10th International Conference Business Information Systems, LNCS 4439, pp.638-651, 2007.

[11] EPCglobal Inc., http://www.epcglobalinc.org
[12] 김의창, 박명수, "시스템 상호 운용성을 위한 웹 서비스 기반의 RFID 미들웨어 구현", 한국정보시스템학회, Vol.18, No.3, pp.71~88, 2009.
[13] 권순량, 이동명, "항만물류 프로세스 분석 및 RFID 적용 기술", 정보처리학회지, Vol.16, No.4, pp.75~84, 2009.
[14] Fosstrak.org, Fosstrak HAL - FEIG configuration (20 Oct. 2008), 16 July 2009 <http://www.fosstrak.org/hal/docs/user-feig.html>
[15] 안재명, 이종태, 오해석 등, EPCGlobalNetwork기반의 RFID 기술 및 활용, Global, 2007.

백선재(Sunjae Baek)

동서대학교 컴퓨터정보공학부

※관심분야 : RFID 애플리케이션 개발, 안드로이드 기반 소프트웨어 개발 등

문미경(Mikyeong Moon)

이화여자대학교 전자계산학과 학사
이화여자대학교 전자계산학과 석사
부산대학교 컴퓨터공학과 공학박사
동서대학교 컴퓨터정보공학부 조교수

※관심분야 : 소프트웨어 공학, 프로덕트 라인 공학, RFID 미들웨어 개발, SOA 기반 소프트웨어 개발 등