韓國컴퓨터情報學會 論文誌
第15卷 第12號, 2010. 12.

2010-15-12-4-4

RFID와 센서 데이터 처리를 위한 EPCIS 저장소 설계 및 구현

현 승 렬*, 이 상 정**

A Design and Implementation of EPCIS Repository for RFID and Sensor Data

Seung-Ryul Hyun*, Sang-Jeong Lee**

요 약

유비쿼터스 컴퓨팅 환경을 구축하기 위하여 자동 식별, 센서 네트워크, 홈 네트워크 등의 다양한 분야에 대한 연구가 진행되고 있다. EPCIS(EPC Information Services)는 RFID 응용 시스템 개발에 필요한 태그 정보를 관리하는 저장소와 관련하여 EPCglobal에서 제안된 표준이다. 본 논문에서는 일반적인 객체 인식과 환경 정보 변화에 의존하는 객체 검색을 가능하게 하기위해 EPCIS 저장소를 설계하고 개발 및 구현한다. 또한 위치를 기반으로 하고 지속적으로 변하는 대용량 자료라는 면에서 RFID 자료와 유사한 센서 데이터를 통합처리 관리하여 USN(Ubiquitous Sensor Network) 환경 변화에 따른 객체 인식 등의 융합 처리가 가능하도록 설계한다.

Abstract

In order to build up the ubiquitous computer environment, there are many researches on automatic identification, sensor networks, and home networks etc. EPCIS (EPC Information Services), which is proposed by EPCglobal, is a standard on the repository managing tag data that is needed to develop RFID application system. In this paper, the EPCIS repository is designed and implemented. It is able to search the object dependent upon general object recognition and environment information variation. And sensor data, which is also massive data and is changed with position, is integrated into RFID data in the system. By doing so, it is possible to do the convergence managements of object recognition with variations of USN (Ubiquitous Sensor Network) environment.

▸ Keyword : Ubiquitous, RFID, Sensor Network, EPCIS, EPCglobal, USN

• 제1저자 : 현승렬
• 투고일 : 2010. 10. 07, 심사일 : 2010. 10. 12, 게재확정일 : 2010. 11. 04.
* 두원공과대학 인터넷정보과 조교수 ** 순천향대학교 컴퓨터학부 교수

Ⅰ. 서 론

유비쿼터스 컴퓨팅 환경을 구축하기 위하여 자동 식별, 센서 네트워크, 홈 네트워크 등의 다양한 분야에 대한 연구가 진행되고 있다. RFID(Radio Frequency IDentification)는 특정 객체에, 정보를 담고 있는 태그를 부착하고, 무선 주파수를 이용하여 리더(Reader)로 객체를 인식하는 시스템을 의미한다. RFID 태그 가격의 하락으로 인해 물류 관리, 재고 관리, 생산관리 등의 다양한 분야에서 RFID 시스템 도입이 확산되고 있다.[1,2,3]

EPCglobal 은 RFID 응용 시스템 개발에 필요한 미들웨어와 저장소 등의 서브시스템에 대한 구현 방법에 대한 표준이 아니라 각 시스템 간의 인터페이스에 대한 표준을 제안한다. 이 표준을 적용해서 시스템을 개발한다면, 시스템을 개발하는 입장에서는 유연하면서 특징 있는 서브시스템을 개발할 수 있게 해 주고, 사용자 입장에서는 어떤 서브시스템을 선택하든지 응용 프로그램의 독립성을 보장한다고 할 수 있다.

EPCglobal 은 미들웨어 관련해서 ALE (Application Level Events)를 그리고, 저장소와 관련해서 EPCIS (EPC Information Services)를 표준으로 제안한다.

USN(Ubiquitous Sensor Network)은 특정 장소에 설치된 센서를 통해 실시간으로 변화하는 각 장소의 온도, 습도, Co2농도 등의 환경 정보를 감지해서 관리 및 통제할 수 있도록 구축한 네트워크이다. USN 구축에 필요한 센서 데이터는 위치를 기반으로 하고 지속적으로 변하는 대용량 자료라는 면에서 RFID 자료와 유사한 면이 있기 때문에, 두 자료가 함께 관리된다면 환경 변화에 따른 객체 인식 등의 융합 처리가 가능하다.[4]

본 논문에서는 일반적인 객체 인식과 환경 정보 변화에 의존하는 객체 검색을 가능하게 하기 위해 EPCglobal에서 표준으로 제안하는 EPCIS 저장소를 RFID와 센서 데이터를 함께 처리할 수 있도록 설계하고 구현한다. 데이터베이스의 설계는 저장 공간의 효율을 위해 테이블을 최대한 분리해서 관계를 설정하고, 질의 형태에 따라 분류해서 뷰(View)를 생성해 놓고 질의문 생성 시에 해당 뷰를 사용하도록 함으로서 질의문 생성을 간략하게 한다. 그리고, 제안 시스템은 서버 측과 클라이언트 측의 모듈로 구분되며, 클라이언트와 서버 간에는 저장소가 자료를 수집하는 역할을 하는 Capturing Service는 Http 프로토콜을 이용하고, 질의 요청에 대한 처리를 담당하는 Query Control Interface 는 Web Service를 사용하며, 비동기 질의 결과 전송을 위한 Callback Interface는 Http를 이용해서 구현한다.

본 논문의 구성은 다음과 같다. 2장은 EPCglobal 표준에 대해 소개하고, 관련 연구에 대한 소개와 제안 시스템에 대하여 비교를 한다. 3장에서는 제안 시스템의 데이터베이스 및 인터페이스에 대한 설계에 대해 기술한다. 4장은 구현 및 적용으로서 구현 내용을 실험하고 수행 결과를 확인한다. 그리고 5장에서는 결론 및 향후 연구 방향에 대해 기술한다.

Ⅱ. 관련 연구

2.1 EPCglobal Standard

이 절에서는 본 논문에서 제안하는 시스템의 대상이 되는 EPCglobal 표준에 대해서 기술한다.[5,6,7,8] RFID 시스템은 태그에 정보를 저장하거나 정보를 읽어오는 리더(쓰기 기능 포함), 리더로부터 필요 정보를 받아 정제해서 저장하는 미들웨어, 미들웨어로부터 정보를 받아 질의 서비스를 제공하는 EPCIS 저장소, 그리고 웹상의 DNS (Domain Name System)와 유사하게 객체의 위치를 제공하는 ONS (Object Name Service) 서버 등의 서브시스템으로 구분해서 구현할 수 있다. EPCglobal 은 RFID 시스템 구축에 필요한 서브시스템의 구현 방법을 제시하는 대신에 서브시스템간의 인터페이스에 대한 표준을 제안한다. 그림1은 EPCglobal에서 제안하는 표준을 보여준다.

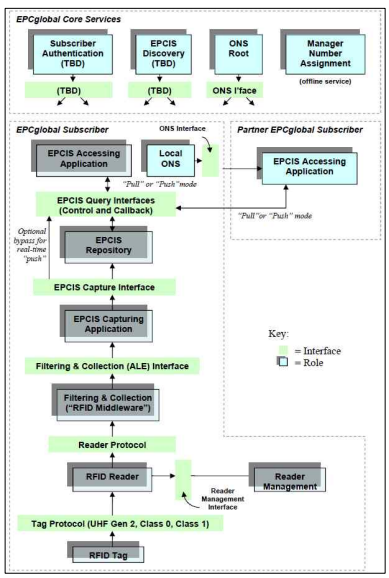

그림 1. EPCglobal 구성 체제
Fig. 1. EPCglobal Architecture Framework

2.1.1 ALE Specification

ALE (Application Level Events) Specification은 RFID 리더로부터 이벤트를 받아서 정제하고 분류해서 응용 프로그램에 전달하기 위한 리더 중립적인 인터페이스를 제공한다.[9,10] ALE 호환 미들웨어를 사용하는 응용 프로그램에서는 각 리더의 장치 드라이버를 설치할 필요가 없으며, 또한 각 리더별로 상이하게 제공되는 별도의 API (Application Program -ming Interface)를 사용할 필요도 없다. ALE는 리더로부터 자료를 읽어오기 위한 Reading API, 태그에 정보를 저장하기 위한 Writing API, 리더의 논리적인 이름을 지정 및 관리하기 위한 Logical Reader API, 그리고 권한을 관리하는 Access Control API 등의 표준을 제안한다.

2.1.2 EPCIS

EPCIS (EPC Information System)은 미들웨어로부터 태그 이벤트 정보를 받아 상품의 상태 및 추적 정보를 생성하여 저장소에 저장하고 관리한다. EPCIS를 사용하는 응용 프로그램은 질의를 요청할 수 있으며, EPCIS 는 저장하고 있는 정보에서 요청한 질의를 수행하고 결과를 반환한다. EPCglobal 은 EPCIS 시스템 구축을 위해 자료를 수집해서 EPCIS에 저장하기 위한 Capture Service, 클라이언트의 질의 처리 및 응답을 위한 Query Control Interface, 그리고 비동기 방식의 결과 제공을 위한 Callback Interface 에 대한 표준을 제공한다. 그림2는 EPCIS 표준을 준수하기 위해 구현해야하는 세 가지 인터페이스를 보여준다. 그리고 이름과 값의 쌍으로 표현된 QueryParam 클래스는 poll 이나 subscribe 같은 질의 요청 메소드에서 질의 요청 조건을 배열로 전달한다.

그림 2. EPCIS 표준 인터페이스와 QueryParam 클래스
Fig. 2. EPCIS Standard Interface and QueryParam Class

2.2 관련 연구

이승주는 EPCglobal 표준을 따르는 EPCIS 시스템을 설계하고 구현하였다.[11] T.Nguyen 등은 EPCIS를 위한 이벤트 기반 저장소와 EPC 질의를 위한 알고리즘을 제안하고 구현하였다.[12] 이들은 EPCglobal에서 제안하는 표준을 따름으로서 RFID 자료 처리에 적합하게 구현되었으나 센서 데이터 처리는 고려하지 않았다.

양문석과 변영철은 다양한 유형의 센서 데이터를 EPC 데이터 형식으로 변환하는 방법을 제안하였다.[4] 이 방법을 이용하면 RFID 자료를 처리하는 기존의 미들웨어에서 센서 데이터를 처리하게 할 수 있다.

김경주 등은 RFID 태그, 센서 및 위치 태그를 포함하는 다양한 정보태그에 대한 통합 연속 질의처리를 지원하기 위하여 확장 이벤트 모델링을 이용해서 센서, 위치, 식별자에 대한 정보의 저장 및 검색을 지원하는 확장 EPCIS의 설계를 제안하였다.[13] 이는 독립적인 서버로 분산 처리되던 센서, 위치, 식별자에 대한 EPCIS 시스템을 하나의 시스템에서 운영할 수 있도록 해주는 통합 시스템이라 할 수 있으며, 각 분산 시스템으로부터 각각 처리된 결과를 네트워크를 통해 수집한 후 최종 결과를 위해 다시 질의를 수행하던 기존 방식보다는 네트워크를 통한 자료 수집에 필요한 시간을 감소시킬 수 있지만, 센서/위치/식별자에 대한 자료를 분리해서 관리하고 별도의 질의 처리를 수행한다.

III. EPCIS 저장소 설계

이 장에서는 제안하는 EPCIS 저장소의 설계에 대해 기술한다. 제안 시스템은 ALE 기반 미들웨어와 통합 가능한 EPCIS 시스템 구축을 목표로 하고 있기 때문에 ALE 기반 미들웨어에서의 사용을 위해 DB 설계 부분을 유연하게 구성한다.

3.1 시스템 구성

그림3은 제안하는 시스템의 전체 구성을 보여준다. ALE로부터 태그 데이터를 수집하는데 사용되는 Capture Service 는 HTTP POST 방식을 통해 EPCIS 저장소로 자료를 전송하며 전송받은 데이터를 DB에 저장한다.

subscribe를 제외한 클라이언트의 동기 방식 질의 요청에 대해서는 Query Interface가 Web Service를 통해 요청을 받아 질의 인수를 분석해서, SQL 구문을 생성하며, 생성된 SQL문을 실행시킨 결과를 클라이언트로 바로 전송한다. 그리고, 비동기 방식으로 작동되는 subscribe 에 대해서는 인수에 의해 생성된 SQL 구문을 클라이언트에서 요청한 주기마다 수행 시킨 후 생성된 질의 결과를 HTTP POST 방식을 통하여 Callback Interface를 통해 클라이언트로 전송한다.

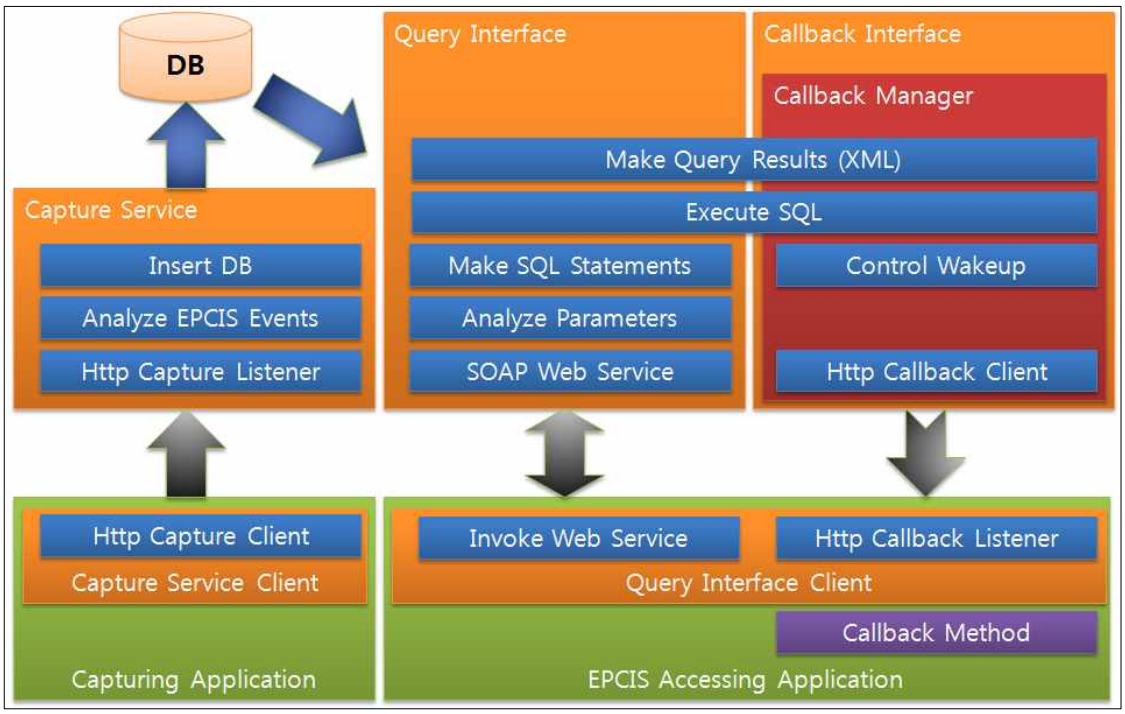

그림 3. 제안 시스템 구성
Fig. 3. Configuration of Proposed EPCIS System

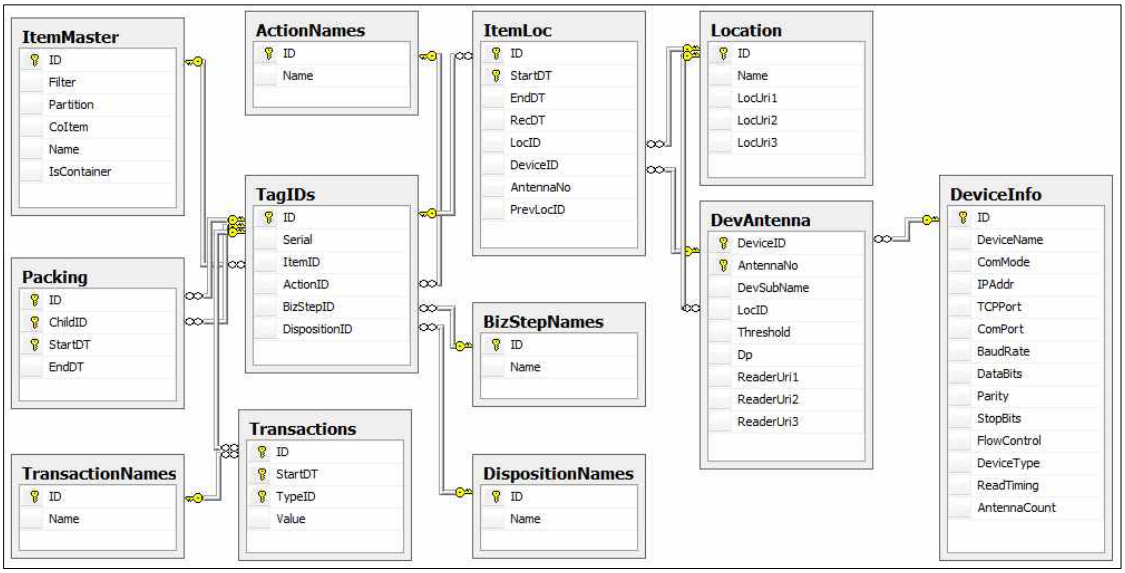

그림 4. 테이블간의 관계
Fig. 4. Relations among the Tables

3.2 데이터베이스 설계

그림4는 본 시스템을 위해 구성한 데이터베이스내의 테이블과 테이블간의 상관관계를 보여준다. 저장 공간의 효율적인 사용을 위해 테이블을 최대한 분리하였으며, 조인을 통해 분리한 테이블을 연결할 수 있다. 그리고, RFID 자료와 유사한 성격을 가지는 센서 데이터의 처리를 가능하도록 구성하였다.[14,15,16,17,18] 표1은 제안한 시스템을 구현하기 위해 설계한 테이블의 목록과 각 테이블에 대한 설명이다.

표 1. 테이블 목록
Table 1. Table List

데이블	설 명
ItemMaster	제품 또는 센서의 기본 정보 제품의 회사 코드와 제품명을 포함
TagIDs	태그 Serial 또는 센서 값 태그 당 한 레코드씩 생성 자료의 중복 방지를 위해 Serial No 만 저장하고, ItemID를 통해 ItemMaster 와 연결
Packing	제품의 포함 관계 포장 안에 소포장 자료를 저장할 수 있는 구조
ItemLoc	제품의 저장 위치를 변경 날짜별로 저장
Location	창고 등의 장소에 대한 Uri를 저장
DeviceInfo	센서 또는 RFID 장치에 대한 정보 ALE 미들웨어에서 장치와의 통신에 사용할 Protocol 지정
DevAntenna	장치에 딸린 측정기나 Antenna 정보 저장(센서 또는 RFID 리더 장치에 따라 하나의 장치에 여러 측정기가 포함되는 경우가 존재) 센서의 경우 DevSubName에 센서의 종류
ActionNames	Action Name 코드 테이블
BizStepNames	Business Step 코드 테이블
DispositionNames	Disposition 코드 테이블
Transactions	제품에 대한 Transaction을 날짜별로 저장
TransactionNames	Transaction 코드 테이블

표2는 본 시스템에서 사용되는 뷰의 목록을 보여준다. 그림4의 테이블간의 관계에 따라 필수적인 조인을 미리 구성해 놓음으로서, SQL 구문 생성 시 Select 구문을 단순하게 해 주며, 질의 작업의 수행 효율을 위해 요청한 질의 종류에 따라 일반 객체 질의, 센서 포함 일반 객체 질의, 트랜잭션 질의, 포함 관계 질의의 네 가지 형태의 뷰 중에서 선택적으로 사용한다.

표 2. 뷰 목록
Table 2. View List

데이블	설 명
V_ItemMaster	회사 식별번호와 객체 번호 구분
V_ObjItem	일반적인 Object Event 질의에 사용
V_ObjItemSensor	센서 정보를 포함한 질의에 사용
V_TransItem	Transaction 질의 요청시 사용
V_AggItem	포함 관계 질의시 사용 (WITH 를 이용해서 Recursive 하게 처리)

뷰 중에서 V_ObjItem 에 설정된 SQL 문은 다음과 같으며, 관계 설정대로 조인한 후 센서 장치를 제외하고 RFID 장치만 선택한다.

```
SELECT TagIDs.ID, V_ItemMaster.Filter, V_ItemMaster.Co,
V_ItemMaster.Item, TagIDs.Serial, ItemLoc.StartDT, ItemLoc.EndDT,
ItemLoc.RecDT, V_ItemMaster.IsContainer, ItemLoc.LocID,
Location.LocUri1, Location.LocUri2, Location.LocUri3,
ItemLoc.DeviceID, ItemLoc.AntennaNo, DevAntenna.ReaderUri1,
DevAntenna.ReaderUri2, DevAntenna.ReaderUri3, TagIDs.ActionID,
ActionNames.Name AS Action, TagIDs.BizStepID,
BizStepNames.Name AS BizStep, TagIDs.DispositionID,
DispositionNames.Name AS Disposition
FROM TagIDs INNER JOIN V_ItemMaster
    ON TagIDs.ItemID = V_ItemMaster.ID
 INNER JOIN DispositionNames
    ON TagIDs.DispositionID = DispositionNames.ID
 INNER JOIN BizStepNames
    ON TagIDs.BizStepID = BizStepNames.ID
 INNER JOIN ActionNames
    ON TagIDs.ActionID = ActionNames.ID
 INNER JOIN ItemLoc
    ON TagIDs.ID = ItemLoc.ID
 INNER JOIN Location
    ON ItemLoc.LocID = Location.ID
 INNER JOIN DevAntenna
    ON ItemLoc.DeviceID = DevAntenna.DeviceID AND
       ItemLoc.AntennaNo = DevAntenna.AntennaNo
 INNER JOIN DeviceInfo
    ON DevAntenna.DeviceID = DeviceInfo.ID
WHERE     (DeviceInfo.DeviceType <> 'S')
```

3.3 센서 데이터 처리

그림5는 EPCIS에서 지원하는 이벤트 타입을 보여준다. 이벤트 타입은 저장소와 응용 프로그램 간에 인수로 주고 받는 자료형으로 볼 수 있으며, 표준 외의 부가적인 정보는 <<Extension Point>> 부분에 추가 가능하다.

센서 데이터에 대한 질의 요청을 위해, 질의 인수 부분에 EQ_Sensor.Type 형식을 추가한다. Type에는 "Temperature", "Humidity", "Co2" 등의 센서 이름을 기술하며, 이 이름은 DevAntenna 테이블의 DevSub Name 컬럼에 이미 저장되어 있어서, 질의 수행 시 SELECT 문의 WHERE 절에서 선택을 위해 사용한다. 질의 처리 결과는 이벤트 타입의 기본 자료와 함께 센서와 관련된 내용을 확장해서 전달하며, XML로 표현하면

```
<Sensor Name="Co2">30.0</Sensor>
```

형식으로 보내준다.

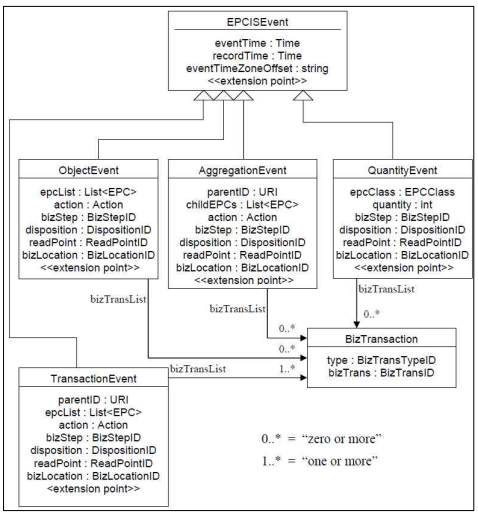

그림 5. EPCIS 이벤트 타입
Fig. 5. EPCIS Event Types

3.4 Capture Service 설계

그림6은 Capture Service를 구현한 클래스 다이어그램을 보여준다. CaptureOperations 클래스는 클라이언트에서 사용할 클래스로서 EPCIS 표준에서 제시하는 EPCISCoreCaptureService 인터페이스를 구현하며, 클라이언트에서 수집한 EPCIS 이벤트 자료를 capture 메소드를 이용해서 서버 측에 전송할 수 있다. CaptureService 클래스는 서버 측에서 이벤트 자료 수신 작업에 사용되며, HTTP Listener 역할을 하는 Capture 클래스를 쓰레드로 실행 시킨다.

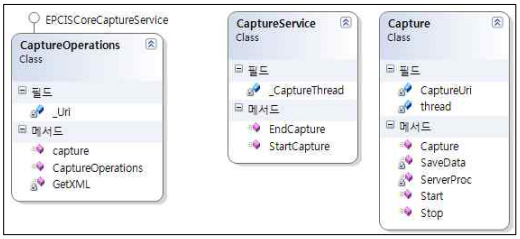

그림 6. Capture Service 클래스 다이어그램
Fig. 6. Capture Service Class Diagram

3.5 Query Interface 설계

그림7은 Query Interface를 구현한 클래스 다이어그램을 보여준다. QueryOperations 클래스는 클라이언트에서 사용할 클래스로서 EPCIS 표준에서 제시하는 EPCISQueryControlInterface 인터페이스를 구현하며, 클라이언트에서 제공되는 메소드를 통해 EPCIS 저장소에서 질의 결과를 가져 올 수 있다. QueryControl Interface 클래스는 QueryInterface 웹 메소드를 Web Service를 통해 노출하며, 클라이언트의 요청에 따라 요청한 수행 결과를 생성해서 전송한다. SQLMaker 클래스는 클라이언트에서 전송한 질의 인수를 분석해서 SelectCommand 클래스를 이용해서 SQL 구문을 생성한다. SelectCommand 클래스는 SQL 구문을 만들고, 구문을 실행시키며, 수행 결과를 생성한다. RFIDTag 클래스는 이진 항목으로 저장된 태그 자료의 Encoding 처리를 수행하며, UriPats 클래스는 Uri 의 표현과 각 항목의 그룹 표현을 관리한다.

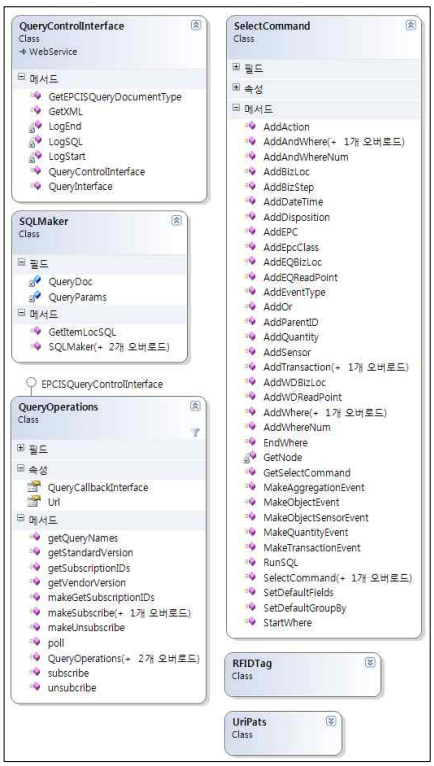

그림 7. Query Interface 클래스 다이어그램
Fig. 7. Query Interface Class Diagram

3.6 Callback Interface 설계

그림8은 Callback Interface를 구현한 클래스 다이어그램을 보여준다. CallbackService 클래스는 클라이언트단에서 HTTP Listener 역할을 하는 Callback 클래스를 쓰레드로 실행 시키며, 서버 측에서 결과 자료가 도착했을 때 클라이언트에 작성해 놓은 Callback 메소드를 호출해줌으로서 결과를 전송한다. SubscribeService 클래스는 비동기 호출에

대한 전체 쓰레드를 관리하며, Subscribe Thread 클래스를 쓰레드로 실행시킨다. SubscribeTh read 클래스는 클라이언트에서 요청한 주기마다 SQL 문을 실행시키고 결과를 전송한다.

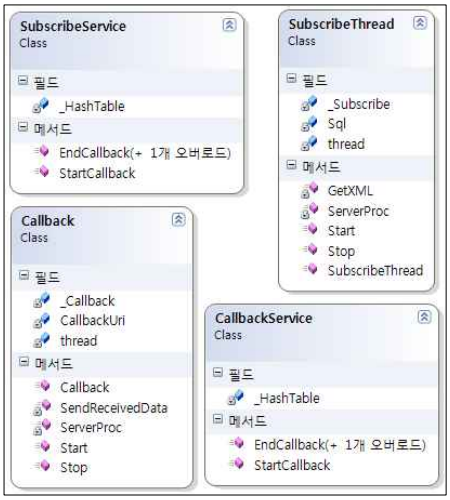

그림 8. Callback Interface 클래스 다이어그램
Fig. 8. Callback Interface Class Diagram

IV. 구현 및 적용

4.1 구현 환경

본 논문에서 제안한 저장소는 MS SQL Server 데이터베이스를 사용하였으며, Visual Studio 2008을 이용해서 C# 언어로 개발했다.[19,20] Query Control Interface 는 Windows 7 OS 에서 제공하는 IIS(Internet Information Server)를 통해 웹 서비스를 제공한다. 실험을 위해 클라이언트에서의 사용의 편의성을 위해 제공하는 Dll(Dynamic Link Library)을 이용해서 테스트 프로그램을 작성하였고, Query Control Interface에서 제공하는 모든 메소드를 수행 가능하도록 하였으며, 이 또한 C# 언어를 이용하여 개발하였다. 실험 프로그램은 실행할 메소드 선택, 메소드 수행에 필요한 질의 인수 입력, 서버로 보낼 자료의 XML 표현, 그리고 서버로부터 질의 처리 결과로 받은 자료에 대한 XML 내용을 확인 할 수 있도록 구성 하였다.

그림9는 실험 프로그램에서 사용할 메소드를 선택하는 화면으로서 질의 인수 이외 메소드 수행에 필요한 인수를 지정할 수 있다.

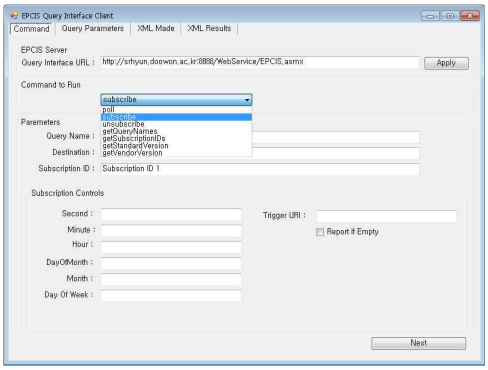

그림 9. 메소드 선택 화면
Fig. 9. A Screen of Method Selection

그림10은 질의 수행에 필요한 인수를 입력하는 화면으로서 EQ_bizTransaction_*Type* 인수와 EQ_disposition 인수를 이용해서 PurchaseOrder 번호가 '1' 이면서 disposition 이 'Not_Sellable' 인 자료를 검색하는 인수를 지정한 화면이다. 화면 윗 부분에서 지정할 인수 종류와 값을 선택하면 지정된 인수는 화면 하단의 목록에 첨부된다.

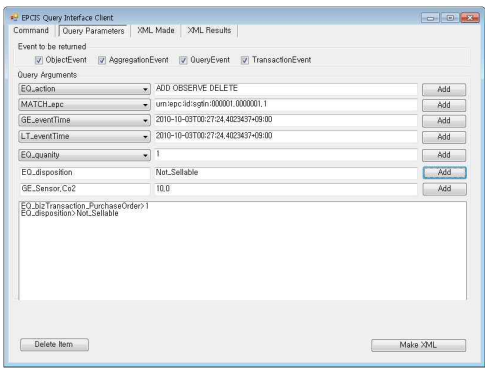

그림 10. 질의 인수 입력 화면
Fig. 10. A Screen of Query Parameter Input

4.2 적용 시나리오

그림11과 같이 2개의 창고를 관리하는 재고 관리에 제안 시스템을 적용한다. 각 창고에는 1개씩의 RFID 리더와 온도, 습도 및 Co_2 농도를 측정할 수 있는 센서가 설치되어 있다. 각 창고에는 개별 물품과 파레트 위에 상자에 포장된 물품이 존재한다. 그리고 개별 물품 및 상자 그리고 파레트에는 그림11에 표시한대로 고유의 RFID 태그가 부착되어 있으며, 파레트와 상자와 물품 간에는 포함관계가 설정되어 있고, 그림11에 표시된 내용 이외에는 Action

에는 'Add', BizStep 은 'Stocking', Disposition 은 'Processing' 그리고 P/O(Purchase Order) 번호는 '9' 로 설정되어 있다.

그림 11. 실험 시나리오
Fig. 11. Test Scenario

4.3 적용 및 결과

실험 프로그램을 이용해서 시나리오에 표현된 창고에 있는 물품에 대한 질의 처리를 확인하기 위해 세 가지 종류의 질의를 요청하고 질의 처리 결과를 확인한다.

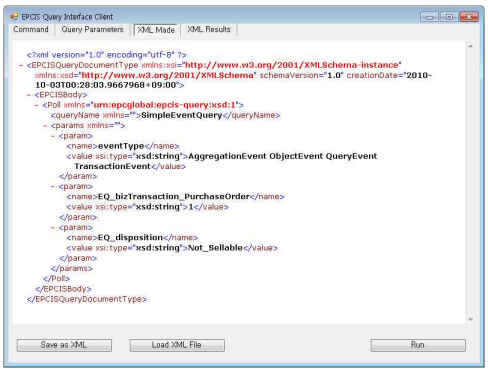

그림 12. 질의 인수 XML (1)
Fig. 12. XML Form of Query Parameter (1)

```
SELECT *
FROM V_TransItem
WHERE
( (TransType = 'PurchaseOrder' AND TransValue = '1' ) )
AND ( Disposition = 'Not_Sellable' )
```

그림 13. 생성된 Select 구문 (1)
Fig. 13. Created Select Statement (1)

```
<?xml version="1.0" encoding="utf-8" ?>
<QueryResults
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <resultsBody>
        <EventList>
            <TransactionEvent>
                <eventTime>2010-01-01T00:00:00</eventTime>
                <recordTime>2010-01-01T00:00:00</recordTime>

<eventTimeZoneOffset>+09:00</eventTimeZoneOffset>
                <bizTransactionList>
                    <bizTransaction type=
                        "PurchaseOrder">1</bizTransaction>
                </bizTransactionList>
                <epcList>
                <epc>urn:epc:id:sgtin:000001.0000003.2</epc>
                </epcList>
                <action>ADD</action>
<bizStep>urn:epcglobal:epcis:bizstep:fmcg:Stocking</bizStep>
<disposition>urn:epcglobal:epcis:disp:fmcg:Not_Sellable</disposition
>
                <readPoint>
                    <id>urn:epc:id:sgln:1.8.1</id>
                </readPoint>
                <bizLocation>
                    <id>urn:epcglobal:fmcg:loc:1.7.1</id>
                </bizLocation>
            </TransactionEvent>
            <TransactionEvent>
                <eventTime>2010-01-01T00:00:00</eventTime>
                <recordTime>2010-01-01T00:00:00</recordTime>
<eventTimeZoneOffset>+09:00</eventTimeZoneOffset>
                <bizTransactionList>
                    <bizTransaction type=
                        "PurchaseOrder">1</bizTransaction>
                </bizTransactionList>
                <epcList>
<epc>urn:epc:id:sgtin:000001.0000003.3</epc>
                </epcList>
                <action>OBSERVE</action>
<bizStep>urn:epcglobal:epcis:bizstep:fmcg:Stocking</bizStep>
<disposition>urn:epcglobal:epcis:disp:fmcg:Not_Sellable</disposition
>
                <readPoint>
                    <id>urn:epc:id:sgln:1.8.2</id>
                </readPoint>
                <bizLocation>
                    <id>urn:epcglobal:fmcg:loc:1.7.2</id>
                </bizLocation>
            </TransactionEvent>
        </EventList>
    </resultsBody>
</QueryResults>
```

그림 14. 질의 결과 XML (1)
Fig. 14. XML Form of Query Results (1)

4.3.1 트랜잭션과 Disposition 질의

제시한 재고관리 시스템에서 "P/O 번호가 '1' 이면서 Disposition 이 'Not_Sellable'" 인 물품을 검색하는 질의를 수행한다. 그림12는 질의를 수행하기 위해 서버 측에 보낼 poll 질의 인수를 보여주며, 질의 조건이 XML로 표현된 것을 확인할 수 있다. 그림13은 서버 측에서 질의 인수를 분석해서 만든 SQL 구문으로서 FROM 절에 V_TransItem 뷰가 지정되었고, 모든 조건이 WHERE 절에 포함되어 있는 것을 확인할 수 있다. 그림14는 서버 측에서 보내준 질의 결과를 XML 형태로 보여주고 있으며, 조건에 맞는 '1.3.2' 물품과 '1.3.3' 물품의 자료가 검색된 것을 확인할 수 있다.

```
<?xml version="1.0" encoding="utf-8" ?>
<EPCISQueryDocumentType
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
schemaVersion="1.0"
creationDate="2010-10-03T00:43:25.2705078+09:00">
    <EPCISBody>
        <Poll xmlns="urn:epcglobal:epcis-query:xsd:1">
            <queryName xmlns="">
                SimpleEventQuery</queryName>
            <params xmlns="">
                <param>
                    <name>eventType</name>
                    <value xsi:type="xsd:string">
                        AggregationEvent
                    </value>
                </param>
                <param>
                    <name>MATCH_parentID</name>
                    <value xsi:type="xsd:string">
                        urn:epc:id:sgtin:000001.0000090.1
                    </value>
                </param>
            </params>
        </Poll>
    </EPCISBody>
</EPCISQueryDocumentType>
```

그림 15. 질의 인수 XML (2)
Fig. 15. XML Form of Query Parameter (2)

4.3.2 포함 관계 질의

포함관계 질의를 확인하기 위해 "파레트 태그 번호가 1.90.1에 포함된 모든 하위 포장과 물품" 을 검색하는 poll 질의를 수행한다. 그림15는 질의를 수행하기 위해 서버 측에 보낼 질의 인수를 보여주며, 질의 조건이 XML안에 표현되어 있는 것을 확인할 수 있다. 그림16은 서버 측에서 질의 인수를 분석해서 만든 SQL 구문으로서 V_AggItem 뷰를 WITH 문으로 재귀 호출한다. 그림17은 서버 측에서 보내준 질의 결과를 XML 형태로 보여주고 있으며, 파레트 안에 2개의 박스가 검색되었고, 각 박스 안에 4개씩의 물품이 검색된 것을 확인할 수 있다.

```
WITH CTEPacking ( ID, Filter, Co, Item, Serial, C_ID,
C_Filter, C_Co, C_Item, C_Serial, C_IsContainer, StartDT,
EndDT, RecDT, LocID, LocUri1, LocUri2, LocUri3, DeviceID,
AntennaNo, ReaderUri1, ReaderUri2, ReaderUri3,
ActionID, [Action], BizStepID, BizStep, DispositionID,
Disposition, [Level] )
AS
(SELECT ID, Filter, Co, Item, Serial, C_ID, C_Filter, C_Co,
C_Item, C_Serial, C_IsContainer, StartDT, EndDT, RecDT,
LocID, LocUri1, LocUri2, LocUri3, DeviceID, AntennaNo,
ReaderUri1, ReaderUri2, ReaderUri3, ActionID, [Action],
BizStepID, BizStep, DispositionID, Disposition, 0
FROM    V_AggItem WHERE ( Co = '000001' AND Item
= '0000090' AND Serial = 1   ) UNION ALL
SELECT A.ID, A.Filter, A.Co, A.Item, A.Serial, A.C_ID,
A.C_Filter, A.C_Co, A.C_Item, A.C_Serial, A.C_IsContainer,
A.StartDT, A.EndDT, A.RecDT, A.LocID, A.LocUri1,
A.LocUri2, A.LocUri3, A.DeviceID, A.AntennaNo,
A.ReaderUri1, A.ReaderUri2, A.ReaderUri3, A.ActionID,
A.[Action], A.BizStepID, A.BizStep, A.DispositionID,
A.Disposition , B.[Level] + 1
FROM V_AggItem AS A INNER JOIN CTEPacking AS B
ON A.ID = B.C_ID )
SELECT * FROM CTEPacking
```

그림 16. 생성된 Select 구문 (2)
Fig. 16. Created Select Statement (2)

```
<resultsBody>
    <EventList>
        <AggregationEvent>
            <eventTime>2010-01-01T00:00:00</eventTime>
            <recordTime>2010-01-01T00:00:00</recordTime>
        <eventTimeZoneOffset>+09:00</eventTimeZoneOffset>
        <parentID>urn:epc:id:sgtin:000001.0000090.1</parentID>
            <childEPCs>
                <epc>urn:epc:id:sgtin:000001.0000091.1</epc>
                <epc>urn:epc:id:sgtin:000001.0000092.1</epc>
            </childEPCs>
            <action>ADD</action>
<bizStep>urn:epcglobal:epcis:bizstep:fmcg:Stocking</bizStep>
<disposition>urn:epcglobal:epcis:disp:fmcg:Processing</disposit
ion>
            <readPoint><id>urn:epc:id:sgln:1.8.1</id>
            </readPoint>
            <bizLocation>
                <id>urn:epcglobal:fmcg:loc:1.7.1</id>
            </bizLocation>
        </AggregationEvent>
        <AggregationEvent>
            <eventTime>2010-01-01T00:00:00</eventTime>
            <recordTime>2010-01-01T00:00:00</recordTime>
        <eventTimeZoneOffset>+09:00</eventTimeZoneOffset>
        <parentID>urn:epc:id:sgtin:000001.0000092.1</parentID>
            <childEPCs>
                <epc>urn:epc:id:sgtin:000001.0000002.1</epc>
                <epc>urn:epc:id:sgtin:000001.0000002.2</epc>
                <epc>urn:epc:id:sgtin:000001.0000002.3</epc>
                <epc>urn:epc:id:sgtin:000001.0000002.4</epc>
            </childEPCs>
            <action>ADD</action>
<bizStep>urn:epcglobal:epcis:bizstep:fmcg:Stocking</bizStep>
<disposition>urn:epcglobal:epcis:disp:fmcg:Processing</disposit
ion>
            <readPoint><id>urn:epc:id:sgln:1.8.1</id>
            </readPoint>
            <bizLocation>
                <id>urn:epcglobal:fmcg:loc:1.7.1</id>
            </bizLocation>
        </AggregationEvent>
        <AggregationEvent>
            <eventTime>2010-01-01T00:00:00</eventTime>
            <recordTime>2010-01-01T00:00:00</recordTime>
        <eventTimeZoneOffset>+09:00</eventTimeZoneOffset>
        <parentID>urn:epc:id:sgtin:000001.0000091.1</parentID>
            <childEPCs>
                <epc>urn:epc:id:sgtin:000001.0000001.1</epc>
                <epc>urn:epc:id:sgtin:000001.0000001.2</epc>
                <epc>urn:epc:id:sgtin:000001.0000001.3</epc>
                <epc>urn:epc:id:sgtin:000001.0000001.4</epc>
            </childEPCs>
            <action>ADD</action>
 <bizStep>urn:epcglobal:epcis:bizstep:fmcg:Stocking</bizStep>
<disposition>urn:epcglobal:epcis:disp:fmcg:Processing</disposit
ion>
            <readPoint><id>urn:epc:id:sgln:1.8.1</id>
            </readPoint>
            <bizLocation>
                <id>urn:epcglobal:fmcg:loc:1.7.1</id>
            </bizLocation>
        </AggregationEvent>
    </EventList>
</resultsBody>
```

그림 17. 질의 결과 XML (2)
Fig. 17. XML Form of Query Results (2)

4.3.3 센서 관련 질의

센서 관련 질의를 확인하기 위해 "창고의 온도가 40도 이상이고 Action 이 'OBSERVE'이며 BizStep 이 'Storing'"

인 물품을 검색해서 1시간에 한번 씩 결과를 반환하는 subscribe 질의를 수행한다. 그림18은 질의를 수행하기 위해 서버 측에 보낼 subscribe 질의 인수를 보여준다. 그림19는 서버 측에서 질의 인수를 분석해서 만든 SQL 구문으로서 FROM 절에 V_ObjItemSensor 뷰가 지정되었고, 모든 조건이 WHERE 절에 포함되어 있는 것을 확인할 수 있다. 그림20은 서버 측에서 보내준 질의 결과를 XML 형태로 보여주고 있으며, 조건에 맞는 '1.5.2' 번 물품이 검색된 것을 확인할 수 있다.

```
<?xml version="1.0" encoding="utf-8" ?>
<EPCISQueryDocumentType
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
schemaVersion="1.0"
creationDate="2010-10-03T17:09:55.2971953+09:00">
    <EPCISBody>
      <Subscribe
xmlns="urn:epcglobal:epcis-query:xsd:1">
            <queryName xmlns="">SimpleEventQuery
            </queryName>
            <params xmlns="">
            <param>
                  <name>eventType</name>
                  <value xsi:type="xsd:string">
                        ObjectEvent</value>
            </param>
            <param>
                  <name>EQ_action</name>
                  <value xsi:type="xsd:string">
                      OBSERVE</value>
            </param>
            <param>
                  <name>EQ_bizStep</name>
                  <value xsi:type="xsd:string">
                  urn:epcglobal:epcis:bizstep:fmcg:Storing
                  </value>
            </param>
            <param>
                  <name>
                        GE_Sensor.Temperature</name>
                  <value xsi:type="xsd:float">40</value>
                </param>
            </params>
            <dest xmlns="">
              http://srhyun.doowon.ac.kr:8888/Callback/
            </dest>
            <controls xmlns="">
               <hour>1</hour>
               <trigger />

<reportIfEmpty>false</reportIfEmpty>
            </controls>
            <subscriptionID xmlns="">
                Subscription ID 1</subscriptionID>
          </Subscribe>
    </EPCISBody>
</EPCISQueryDocumentType>
```

그림 18. 질의 인수 XML (3)
Fig. 18. XML Form of Query Parameter (3)

```
SELECT *
FROM V_ObjItemSensor
WHERE ( Action = 'OBSERVE' )
AND ( (DevSubName = 'Temperature' AND SensorData
>= 40) )
AND ( BizStep = 'Storing' )
```

그림 19. 생성된 Select 구문 (3)
Fig. 19. Created Select Statement (3)

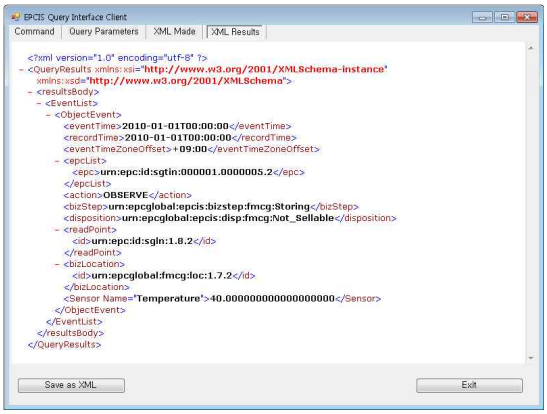

그림 20. 질의 결과 XML (3)
Fig. 20. XML Form of Query Results (3)

4.4 관련 연구와 비교

제안 시스템은 EPCglobal에서 제안하는 표준을 따르면서 RFID와 센서 데이터를 같은 테이블에 저장함으로서 저장 공간을 효율적으로 이용할 수 있으며, 질의 수행의 효율을 높일 수 있고, 또한 센서 자료를 포함한 환경 질의를 수행 할 수 있도록 설계 및 구현하였다. 표3은 관련 연구와 제안 시스템의 비교 내역을 보여준다.

표 3. 관련 시스템과의 비교
Table 3. Comparisons with Other Related System

비교 항목	EPCIS 표준 시스템	EPC 변형을 통한 EPCIS 표준 시스템	통합 연속 질의 시스템	제안 시스템
EPC 자료 처리	○	○	○	○
센서 자료 처리	X	○	○	○
자료 관리	EPC	EPC 또는 센서	EPC, 센서, 위치 자료 별도 보관	통합 보관
일반 질의	○	○	○	○
센서 포함 질의	X	X	△	○
관련 참고 문헌	[11],[12]	[11]+[4], [12]+[4]	[13]	

V. 결론 및 향후 연구

본 논문에서는 일반적인 객체 인식과 환경 정보 변화에 의존하는 객체 검색을 가능하게 하기 위해 EPCglobal에서 표준으로 제안하는 EPCIS 저장소를 RFID와 센서 데이터를 함께 처리할 수 있도록 설계하고 구현 하였다. 또한, 구현한 시스템을 두개의 창고를 관리하는 재고관리에 적용하여 트랜잭션과 Disposition 질의, 포함 관계 질의, 센서 관련 질의에 대한 실험을 수행하고 질의 결과를 확인하였다.

본 논문에서 제안한 EPCIS 저장소는 EPCglobal에서 제안하는 EPC Information Services (EPCIS) Version 1.0.1 Specification 표준을 기반으로 설계 및 구현하였으므로 표준에 맞게 개발한 응용 프로그램에서 시스템의 변경 없이 사용할 수 있는 장점을 가진다. 또한, RFID 자료와 유사한 성격을 가지는 센서 데이터의 처리도 가능할 수 있도록 구현하였기 때문에 특정 장소의 센서 값의 변화에 따른 질의를 처리할 수 있는 장점이 있다.

향후에는 본 논문에서 구현한 EPCIS 시스템을 Java 응용 프로그램에서 구동할 수 있도록 하기 위한 Java 클라이언트 클래스를 구현할 예정이며, 제안한 저장소에 EPCglobal의 ALE (Application Level Interface)에 기반하여 센서 데이터를 처리할 수 있는 미들웨어를 추가함으로서 ALE와 EPCIS 통합 미들웨어 저장소를 구축할 예정이다.

참고문헌

[1] 신명숙, 홍성표, 이준, “RFID/EPC-IS 네트워크를 이용한 제품 추적 및 인증시스템 구현,” 정보처리학회논문지, 제 13-A권, 제 4호, 317-322쪽, 2006년. 8월.

[2] 최원용, 이종태, “MES개선을 위한 RFID 적용 -EPCIS 확장모형을 중심으로-,” 한국콘텐츠학회논문지, 제 7권, 제.12호, 333-345쪽, 2007년. 12월.

[3] 장성호, 마용범, 노창현, 박량재, 김교현, 채홍석, 이종식, 김재영. “유비쿼터스 환경을 위한 RFID 기반의 항공 물류 사스템의 설계 및 구현,” 한국컴퓨터정보학회논문지, 제 12권, 제 6호, 297-306쪽, 2007년. 12월.

[4] 양문석, 변영철, “RFID 미들웨어 기반 센서 데이터 스트림 처리 방법,” 한국해양정보통신학회논문지, 제 12권, 제 2호, 231-239쪽, 2008년. 2월.

[5] EPCglobal, *http://www.epcglobalinc.org/*

[6] EPCglobal, "EPCglobal Tag Data Standards Version 1.4," June 2008.

[7] EPCglobal, "EPC Information Services (EPCIS) Version 1.0.1 Specification," September 2007.

[8] EPCglobal, "The Application Level Events (ALE) Specification, Version 1.1.1," March 2009.

[9] 홍연미, 변영철, “ALE 기반 RFID 미들웨어 설계 및 구현,” 한국해양정보통신학회논문지, 제 11권, 제 4호, 648-655쪽, 2007년. 4월.

[10] 김의창, 박명수, “시스템 상호 운용성을 위한 웹 서비스 기반의 RFID 미들웨어 구현,” 한국정보시스템학회, 정보시스템연구, 제18권, 제3호, 71-88쪽, 2009년 9월

[11] 이승주, “RFID 데이터 질의 처리를 위한 EPCIS 시스템의 설계 및 구현,” 부산대학교대학원 석사학위논문, 2007년 8월

[12] T. Nguyen, Y. K. Lee, B. S. Jeong, S. Lee, "Event Query Processing in EPC Information Services," Third International IEEE Conference on SITIS, pp. 159-166, 2007.

[13] 김경주, 안성우, 홍봉희, “센서, 위치, 식별자를 위한 통합 연속 질의 처리 기법,” 한국정보과학회 학술대회논문집, 제 36권, 제 1호, 175-180쪽, 2009년. 7월.

[14] 이수안, 김진호, 신성현, 남시병, “유비쿼터스 센서 네트워크에서 스트림 데이터를 효율적으로 관리하는 저장 관리자 구현,” 전자공학회 논문지, 제 46권, CI편, 제 3호, 24-33쪽, 2009년. 5월

[15] 복경수, 조용준, 여명호, 유재수, “대용량 RFID 데이터를 위한 효율적인 데이터 관리 기법,” 한국콘텐츠학회논문지, 제 9권, 제 6호, 25-36쪽, 2009년. 6월.

[16] F. Wang and Peiya Liu, "Temporal Management of RFID Data," Proc. International Conference on VLDB, pp. 1128-1139, 2005.

[17] H. Gonzalez, J. Han, X. Li, and D. Klabjan, "Warehousing and Analyzing Massive RFID Data Sets," Proc. International Conference on Data Engineering, pp. 83-83, 2006.

[18] D. Lin, H. G. Elmongui, E. Berino, and B. C. Ooi, "Data Management in RFID Applications," Proc. International Conference on Database and Expert Systems Applications, pp. 434-444, 2007.

[19] 우재남, “뇌를 자극하는 SQL Server 2008,” 한빛미디어, 2009년. 3월.

[20] Microsoft, MSDN Library for Visual Studio 2008.

저 자 소 개

현 승 렬

1988 : 한양대학교 공학사.
1991 : 한양대학교 공학석사.
2009 - 현재 : 순천향대학교 컴퓨터학과 박사과정
2002 - 현재 : 두원공과대학 인터넷정보과 조교수
관심분야 : 데이터베이스, 객체지향프로그래밍, 센서네트워크, RFID

이 상 정

1983 : 한양대학교 공학사.
1985 : 한양대학교 공학석사.
1988 : 한양대학교 공학박사
1988 - 현재 : 순천향대학교 컴퓨터학부 교수
관심분야 : 멀티 코아 프로세서, 모바일 시스템의 성능 및 전력 동적 최적화, IT 융합