USN/RFID 환경에서 상황인식이 가능한 통합 미들웨어 System 설계

한수° 박상현 최용식 전영준 신승호
인천대학교 컴퓨터 공학과
{pucktan°, tank1862, mars, 0961144, shin0354}@incheon.ac.kr

Design of Integrated Middleware System to enable context-aware for USN/RFID Environment

Soo Han° Sang Hyun Park, Yong Sik Choi, Young Jun John, Seung Ho Shin
Dept. of Computer Engineering, University of Incheon

요 약

유비쿼터스 환경을 실현하기 위한 핵심 기술로 RFID가 주목받고 있고 많은 연구가 이루어지고 있다. RFID는 점점 소형화되고 주변의 모든 환경 및 사물에 부착 될 것이다. 이러한 환경에서 RFID Tag가 부착된 개체들의 관리와 Middleware의 중요성이 부각되고 많은 기관에서 연구가 진행되고 있다. USN/RFID 환경 하에서 사용하기 위한 Middleware는 다음과 같은 기능이 필요하다. 첫째, RFID가 부착 된 개체가 주위 환경에 민감한 개체일 경우 Middleware에서 자체적으로 상황을 인지하여 대처해야 한다. 둘째, 방대한 양의 raw 데이터 처리를 부하 없이 처리해야 한다. 셋째, 실제 RFID를 적용했을 경우 한 종류의 RFID 장비만을 사용하는 것이 아니므로 다양한 종류의 RFID 장비를 인식할 수 있는 Middleware가 필요하다.

본 논문에서는 RFID에 센서와 무선통신 기술을 결합하여 상황인식(context-aware)이 가능하고, 분산처리 환경을 도입하여 방대한 양의 데이터를 처리 가능하며, 다양한 프로토콜이나 주파수의 장비를 통합적으로 적용 가능한 Middleware의 설계를 제안한다.

1. 서 론

RFID (Radio Frequency Identification)는 유비쿼터스 세상을 맞이하며 가장 부각되는 기술 중에 하나이다. RFID는 라디오파를 사용하여 특정 물체를 인식하는 모든 기술을 말한다. RFID에 관한 많은 연구가 진행되면서 RFID에서 수집 된 데이터들을 효율적으로 사용하기 위해서 RFID Middleware에 초점이 모아지기 시작했다. RFID Middleware 플랫폼의 기본적인 요구사항은 다음과 같다[1].

첫째, 시장에 존재하는 다수의 이기종 멀티 프로토콜 기반 RFID 리더 사이에 존재하는 이질성을 상위계층에 숨기기 위해 공통의 인터페이스를 정의하고, 이를 통해 태그데이터 수집, 리더 설정, 개별 리더의 모니터링 및 원격제어 등의 관리가 이루어 질수 있도록 기능이 제공되어야 한다.

둘째, 일반적으로 RFID 태그 데이터는 리더로부터 초당 적게는 수십 회부터 많게는 수백회의 대용량의 데이터가 빠른 속도로 System으로 유입되며, 동일한 태그 데이터가 반복적으로 인식된다. 이러한 인식된 태그 정보 전부가 애플리케이션에 요구되지 않기 때문에, 중복 데이터 혹은 불필요한 데이터를 선별하여 제거하는 필터링 기능이 필요하게 된다.

셋째, RFID 리더로부터 수집된 태그 데이터를 데이터 수요자인 WMS (Warehousing Management System : 창고관리 시스템), ERP (Enterprise Resource Plan : 전사적 자원 관리 시스템) 등의 기존 응용System으로 전달하는 기능을 Middleware에서 지원해야 한다.

RFID Middleware 플랫폼의 기본적인 요구사항 이외에 Middleware가 RFID 태그가 부착 된 개체의 상태를 모니터링 하며 개체의 상태에 따라 동작을 취하며 개체 관리도 가능 해진다면 유비쿼터스 환경에 걸 맞는 RFID System이 될 것이다.

본 논문에서는 기존에 연구하던 다중인식 (Multi - aware) RFID Middleware를 통해 다양한 주파수와 프로토콜을 가진 RFID장비들을 통합적으로 수용하고, 기존 연구를 확장하여 리더로부터 수신되는 방대한 raw데이터들을 분산 처리 환경을 도입하여 부하를 줄임으로서 RFID Middleware의 기본적인 요구사항을 충족시키고, RFID에 센서와 무선 통신 기술을 결합하여 상황인지를 가능하게 하여, RFID 태그가 부착 된 개체들을 관리하게 하도록 하는 RFID Middleware System을 설계 한다.

2. 관련 연구

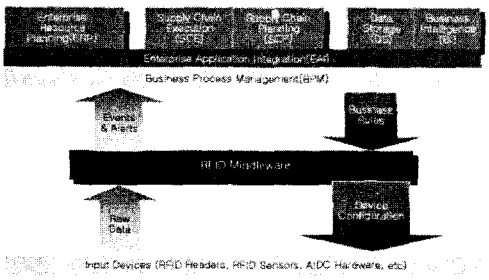
2.1 유비쿼터스 환경에서의 상황인지 컴퓨팅 System

유비쿼터스 시대의 응용 및 서비스는 컴퓨팅 및 커뮤니케이션 능력을 가진 스마트 객체들이 동적인 환경 변화를 인식하고 이에 적응할 수 있는 특성, 즉 상황인식 (context-aware) 특성을 가지고 있어야 한다. 상황정보는 사용자가 상호 작용을 하는 시점에 가용한 거의 모든 정보이다[2]. 또한, 상황정보는 응용 운용 환경의 일부

로 응용이 감지할 수 있는 정보를 포함한다. 이는 일반적으로 사람, 그룹, 객체의 위치, 식별, 활동, 상태 등을 포함한다[3]. 상황인식 서비스는 이러한 상황정보의 수집 및 교환을 통해 인식하고, 해석 및 추론과 같은 처리 과정을 거쳐 사용자에게 상황에 적절한 서비스를 제공한다. 상황정보는 사용자 인터페이스 또는 센서, 센서 네트워크 등을 통해 수집된다. 사용자는 키패드나 터치스크린 등과 같은 사용자 인터페이스를 이용하여 자신의 기본적인 개인 정보나 개인 일정 등과 같은 정적인 상황정보를 입력할 수도 있다. 온도, 습도와 같은 환경적 상황정보와 사용자의 체온, 혈압 등과 같은 정보들은 사용자 단말에 부착된 센서를 통해 직접 수집될 수도 있고, 사용자 주변의 센서 네트워크 또는 상위계층 네트워크와 통신을 통해 수집될 수도 있다. 사람이나 사물 등과 같은 객체의 식별정보도 위치정보와 마찬가지로 상황인식 서비스를 위해 기본적으로 요구되는 상황정보이다.

2.2 RFID Middleware System

[그림 1]에서 보는 봐와 같이 RFID Middleware는 기본적으로 RFID 리더 등과 같은 자동식별 장치와 기업환경의 System들 간의 연결 고리가 되는 부분이다. 따라서 RFID가 다양한 응용 서비스에서 사용되기 위해서는 RFID 리더 및 각종 센싱 디바이스에서 발생하는 방대한 raw data를 실제 의미있는 정보로 재구성하여 응용 서비스가 관심을 가지는 데이터 만을 필터링하여 전달하도록 함으로써 그 데이터 량을 줄이고 이를 적절한 장소와 적절한 시간에 응용 서비스로 전달하는 기능이 요구된다. 데이터 필터링 기능은 데이터의 형식 및 응용 환경에 따라 요구되는 기능이 달라질 수 밖에 없다. 또한, RFID Middleware는 지속적으로 끊임없이 입력되는 방대한 데이터를 정확하게 실시간으로 처리하고 응용 서비스에서 요구하는 결과를 획득해서 전달하여야 하기에, 대량의 데이터 스트림을 적절히 소화해내는 안정성도 함께 요구된다[4].



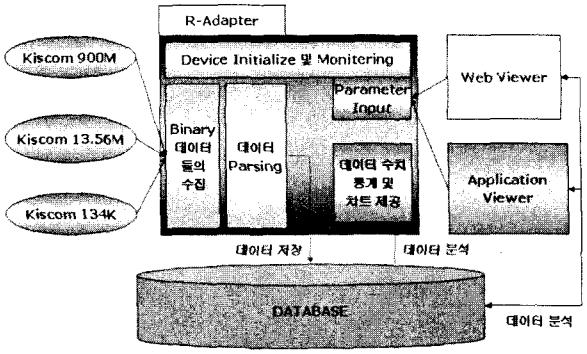
[그림 1] 기업 전산 환경내에서 RFID Middleware의 위치

2.3 Multi RFID-Middleware System

RFID System이 실제로 적용되는 분야에서 보면 단순히 한 종류의 주파수 대역이나 프로토콜을 사용하는 RFID 장비만을 사용하지 않는다. 대표적으로 물류분야에서는 다양한 주파수의 RFID 장비를 필요로 한다.

Multi-aware RFID Middleware은 각각의 서로 다른 프로토콜과 주파수를 가진 장비들에서 바이너리 데이터를 수집하여 각각의 장비에 맞는 형식의 코드로 파싱하고 DB 서버에 저장하여, 데이터의 통계와 분석에 있어서 편이점을 제공한다[5].

[그림 2] 은 Multi-aware RFID Middleware의 설계 화면이다.

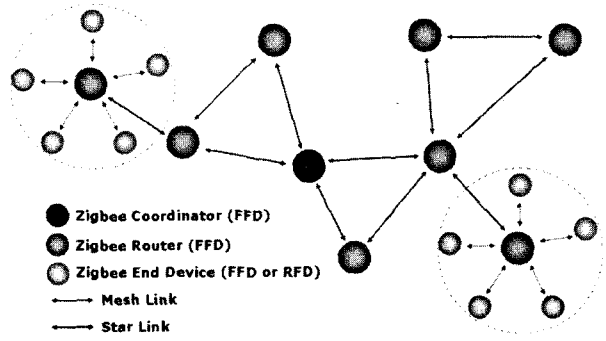


[그림 2] System 설계

2.4 Zigbee

IEEE 802.15.4 Zigbee는 저속, 저가, 저 전력 소모를 필요로 하는 응용에 주안점을 둔 근거리 무선 통신 기술(네트워크 당 255개의 노드 연결)이다. Zigbee는 2.4GHz, 868MHz, 915 MHz의 주파수 대역을 사용하며 각각 250kbps, 20kbps, 40kbps의 데이터 전송률을 제공한다[6].

Zigbee의 MAC 계층은 저 전력 소모를 위한 방식들을 제공하고 있는데, Superframe 구조로 동작하는 방법, Data request frame을 사용하는 방법, backoff 횟수를 줄이는 방법, short address를 사용하는 방법 등으로 이를 실현하고 있다. PHY 계층은 간단한 구조로 되어 있다. 별도의 channel coding 기법을 사용하지 않고, Spreading과 PSK modulation만을 하여 전송하는 구조로 되어 있다. 따라서 근거리의 저속 무선 통신에 한정된 용도를 지녔지만, 낮은 가격으로 실현될 수 있다[7].



[그림 3] Zigbee 네트워크

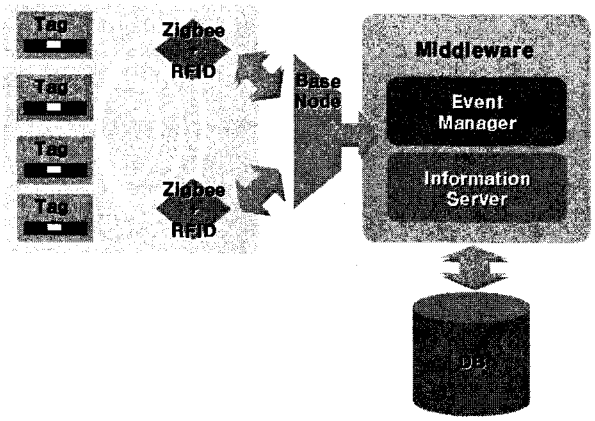
Zigbee는 스타형 및 메시 토폴로지와의 조합을 가능하게 하는데, 이 조합은 클러스터 트리 네트워크라 불린다. 각 네트워크는 초기화, 노드 관리, 노드 정보 저장 기능을 제공하기 위해 코디네이터라고 불리는 FFD (full-function device)가 하나 이상 있어야 한다.

비용과 전력 소모를 최소화하기 위해 나머지 노드는 배터리로 동작하는 간단한 RFD(reduced-function device)로 구성된다. Zigbee 네트워크는 몇몇 데이터 전송 시나리오에 적용된다. 무선 센서 데이터 등의 주기적인 데이터의 경우 노드는 설정 된 횟수만큼 깨어나 샘플링된 데이터를 코디네이터에 전송하고, 다시 절전 모드 상태가 된다. 그러므로 배터리 수명을 연장 시킬 수 있다[8].

3. 제안 System

3.1 개 요

본 논문에서 제시하는 미들웨어는 다음과 같은 System 구성을 가진다.

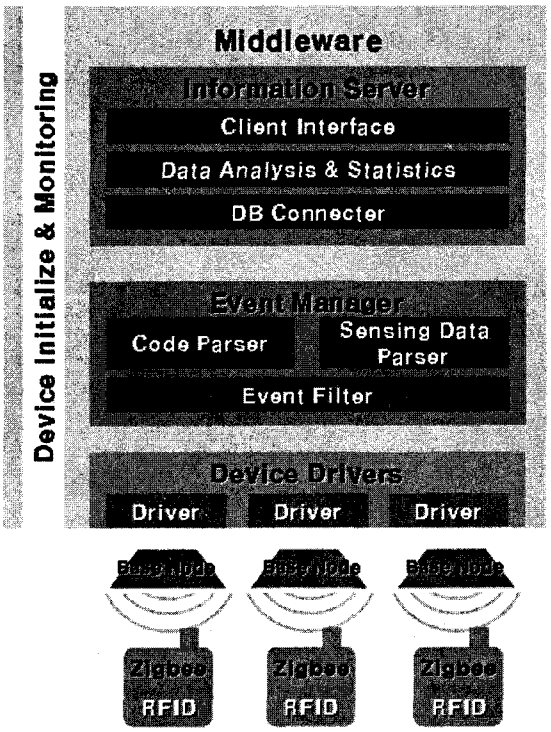


[그림 4 Middleware System 전체 구성도]

개체들에 부착 된 개개의 Tag들을 RFID를 통해 인식하고, 인식 된 데이터들을 Zigbee를 사용하여 Middleware 서버에 전송한다. Middleware는 Event Manager와 Information Server 가 존재 한다. Event Manager는 RFID와 Zigbee 모듈에서 전송받은 이벤트 데이터 들에 대한 관리를 하고, Information Server는 Event Manager을 통해 전송 받은 데이터들을 Database에 저장하고, Database에 저장 된 데이터들을 통해서 분석 결과와 통계치를 제공한다. 또한 Information Server에서는 Clint의 Application에서 데이터를 사용가능하도록 하는 Client Interface를 제공한다.

3.2 Middleware 상세 설계

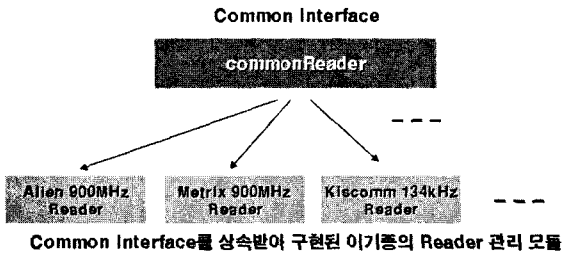
Nicholas D. Evans의 'Middleware is the Key to RFID'에 따르면 Software 측면에서의 RFID 솔루션은 RFID 리더로부터 유입되는 태그 데이터를 의미있는 정보로 변환하여 기존 legacy System과의 연계를 수행하는 미들웨어 역할을 담당하며, 이는 다시 RFID Middleware와 기존 System 통합 미들웨어로 나뉘어진다[9]. RFID System 도입에 따라 RFID 리더와 Middleware간의 연결을 지원하는 새로운 소프트웨어 System이다. 반면 기존 System 통합 Middleware는 조직 내외에 분산되어 있는 legacy System 간의 연계 및 통합을 지원하며, RFID 기술의 도입에 따라 단순하게는 Middleware로부터 전달된 의미있는 정보를 다양한 프로토콜에 기반한 기존 System으로 라우팅하고, 적절한 포멧으로 변환하여 전달하는 역할을 담당하게 된다. 제안하는 RFID Middleware System의 RFID Middleware는 이미 서론에서 거론하였던 RFID Middleware 플랫폼의 기본적인 요구사항을 충족시킨다.



[그림 5 Middleware System 상세 모듈별 구성도]

[그림 5]는 제안하는 RFID Middleware 의 상세 구성도이다. 시장에 존재하는 다수의 이기종 멀티 프로토콜 기반 RFID 리더들에 유연성을 가지는 구조를 위해 기존의 연구인 'U-Logistics환경 구축을 위한 다중 RFID 미들웨어 System의 설계 및 구현[5]' 를 이용하여 리더들의 공통의 인터페이스를 정의하고, 인터페이스의 상속을 통해 개별 리더들의 관리 모듈을 만들고 이 모듈을 통해 태그데이터 수집, 리더 설정, 개별 리더의 모니터링 및 원격제어 등의 관리가 이루어 질수 있도록 하였다. 아래 [그림 6]은 개별 리더 관리모듈들이 공통 인터페이스에

서 상속을 받아 구현 되는 구성을 보여주고 있다.



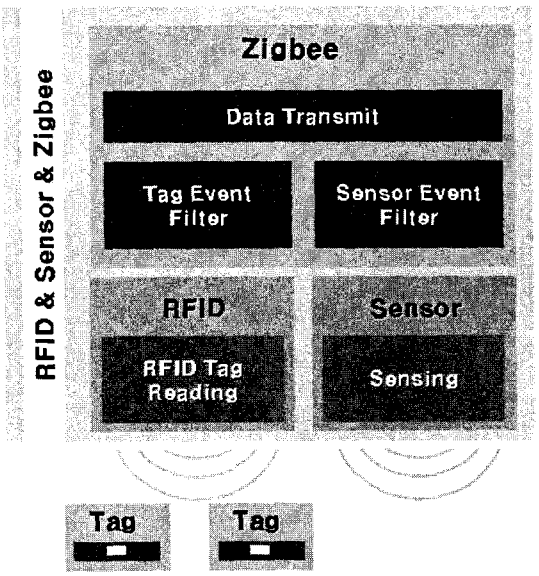
[그림 6 리더 공통 Interface 상속도]

개개의 리더 관리 모듈을 통해 발생한 RFID Tag Event와 Sensor Event는 Event Manager로 전송되며, Event Manager 안의 Event Filter를 통해 중복되어 수신 된 데이터와 의미없는 데이터들을 Fitering한다. 이때, Event Filter에는 RFID Tag Event만이 수신된다. Sensor Event는 Zigbee 모듈에서 필요한 Filtering이 수행되므로, Middleware에서 Filtering이 필요 없기 때문이다. Filtering된 RFID Code 데이터 들은 Tag Code Parser로 이동한다. 마찬가지로 Zigbee에서 Filtering된 Sensor 데이터는 Sensor Parser로 이동한다. 각 Parser에서는 수신 된 Binary 데이터를 RFID Tag 데이터의 경우 리더마다의 코드 표준에 맞추어 Parsing 작업을 수행하고, Sensor 데이터의 경우 해당 Sensor 표준에 맞추어 Parsing 작업을 수행한다. Parsing 되어진 데이터들은 Middleware Buffer에 쌓이고, 수신 시간과 함께 Event Logger에 기록된다. Event Manager에서는 EPC Gen2, ISO, 기타 태그(Active/Passive)와 센서를 지원한다.
Middleware Buffer에 쌓여진 Parsing이 끝난 Code 데이터들과 Sensor 데이터들은 일정한 시간을 기준으로 Information Server로 전송되어지고 Information Server의 DB Connecter에서는 DB와 연결 관리를 수행하고 수신 되어진 데이터들을 저장하고, 필요에 따라 DB에 저장 된 데이터들의 검색 동작을 수행하게 된다. Data Analysis & statistics 에서는 DB에 저장 된 Tag Code 데이터의 인식시간, 빈도수, 기타 정보를 토대로 통계자료를 산출하고, 사용자 정의에 의한 분석 자료를 생성한다. Client Interface에서는 Client Application 에서 요구하는 RFID Tag 데이터와 Sensor 데이터 또는 RFID Tag Event가 발생 시 생성 되는 데이터들을 XML/Http, JMS(Java Message Service), RMI 와 같은 형태로 생성하고 Client Application과 통신하여 데이터의 송수신을 담당하게 된다.

3.3 RFID와 Zigbee 결합 모듈 설계

RFID와 Zigbee 의 결합 모듈은 [그림 7]에서 RFID와 Zigbee 센서 모듈과의 결합 구성도를 보여주고 있다.

RFID와 Zigbee 모듈에서 수집하는 데이터의 경우는 RFID Tag 데이터와 Sensor 데이터 두 가지로 나누어진다. 먼저 RFID Tag 데이터의 경우 RFID 리더를 통해 Tag Event가 들어오면, Tag Event Filter로 이동되어지며, Filter안의 Buffer에 쌓이게 된다.



[그림 7 RFID와 Zigbee 결합 모듈 구성도]

Zigbee내의 Event Filter가 존재하는 이유는 Middleware에서 한꺼번에 이루어지는 Filtering의 부하를 줄여주기 위해서 각 Zigbee node에서 filtering를 수행함으로서 Middleware에서의 부하를 줄여줄 수 있다. Buffer에 쌓인 데이터들은 다음 알고리즘에 의해 Data Transmit 모듈로 이동하게 된다.

```
while true do
    BR <- getTagEvent()
    if (timer()-TB)=TF or checkBufferFull(BR) then
        Transmit(BR)
        ClearBuffer(BR)
        TB <- timer()
    end if
    filter(BR)
end while
```

B_R : RFID Tag Event Buffer
getTagEvent() : tag event를 얻어옴
T_F : 각 리더기의 태그 인식 시간 간격
T_B : Buffer의 초기화 후 시간을 Reset
timer() : Zigbee timer를 이용하여 현재 time을 얻어옴
checkBufferFull(B) : Buffer가 full 상태인지 체크
ClearBuffer(B) : 해당하는 Buffer를 Clear 시킴
transmit(B) : Buffer를 Data Transmit 모듈로 전송
filter(B) : event들의 필터링(중복, dummy 데이터 제거)

Sensor 데이터의 경우에도 RFID Tag 데이터와 마찬가지로 동작하지만 이 경우에는 Event가 들어오게 되면, Sensor Event Filter로 이동되어지고 Buffer에 쌓이기 전

에 상황인식 동작을 위해 일정 센서 한계치와 데이터와 비교하는 동작이 이루어진다. 다음은 Sensor Data를 Transmit 모듈에 보내기 위한 동작이다.

```
while true do
   tems <- getSensorEvent()
   if tems > conditions
      then transmit(tems ,TCA)
      else
         Bs <- tems
         if (timer()-TB)=Ts or checkBufferFull(Bs)
               then transmitBs,OM)
               ClearBuffer(Bs)
               TB <- timer()
          end if
      filter(Bs)
   end if
end while
```

tems : Sensor event를 임시로 담아두는 변수
getSensorEvent() : Sensor event를 얻어옴
conditions : 센서 데이터 제약치
transmit(B,O) : 옵션을 가지고 Sensor 데이터 들을 Data Transmit 모듈로 전송함

온도를 감지하는 센서라고 가정했을 때, 온도가 일정 한계치를 넘게 되는 것을 센서가 감지하면 센서는 이벤트를 Middleware와 함께 context-aware Actor로 이벤트를 보내준다. context-aware Actor는 상황인식에 따라 동작해야 하는 Process를 가진 모듈이다. Middleware에서는 일반적인 event process를 진행하며, context-aware Actor에서는 센서에서 전달받은 event를 통해 인터럽트 프로세스를 수행하여 상황에 따른 동작을 수행하게 한다. Data Transmit모듈에서는 Middleware로 RFID Tag Event와 Sensor Event를 전송하고, Sensor interupt Event를 Context-aware Actor로 전송해주는 역할을 한다.

4. 결 론

유비쿼터스 컴퓨팅에 기반한 제반환경을 구축 하는데 있어서 Middleware는 유비쿼터스 System의 성능을 크게 좌우하고 있다. 많은 유비쿼터스 Middleware System의 연구들이 진행되고 있지만, 유비쿼터스 환경에서 기본적으로 요구하는 사항들을 충족시키면서 상황인식까지 가능한 통합적인 System 설계 및 구축이 미흡다. 그리고 유비쿼터스 Middleware 개발에 있어서 방대한 raw 데이터의 처리는 중요한 문제점이다.

본 논문에서는 유비쿼터스 제반 환경 구축을 위하여 유비쿼터스 Middleware로서 기본적으로 요구하는 사항들을 충족시키면서 RFID와 센서를 결합하여 상황인식까지 가능하게 하는 통합적인 Middleware System의 설계를 제안하였다. 또한 Zigbee에 Event Filter 모듈을 통해서 Middleware의 데이터 처리로 인한 부하를 줄여주었다. 제안되어진 미들웨어가 산업적인 측면에서 사용 되어 진다면 효용성이 높아질 것이다. 예를 들어 식료품 같은 경우는 주위 온도에 민감한 제품인데 식료품 코너에 리더가 설치되어 있고 고객이 제품을 리더에 찍어 정보를 볼 수 있는 System이 설치되어 있다면, RFID와 Zigbee 결합 모듈은 항상 식료품 코너의 주위 온도를 체크하여 온도의 조절이 가능해지고, 다량의 데이터 유입 시에도 Zigbee에서 Filtering을 한 번 더 해줌으로서 Middleware의 부하를 크게 줄일 수 있다.

현 Middleware System에서도 개인 프라이버시와 같은 보안 모듈이 포함되지 않았다. 향후 보안 모듈까지 포함된 Middleware System에 대한 연구가 필요하다.

5. 참고 문헌

[1] 정태수, 김영일, 이용준, "RFID 미들웨어 플랫폼 기술," Telecommunications Review 제15권 2호, 2005. 4

[2] Bill Schilit, Norman Adams and Roy Want, "Context-aware computing applications," In proceedings of IEEE Workshop on Mobile Computing Systems and Applications, 1994.

[3] Guanling Chen, David Kotz, "A survey of context-aware mobile computing research," Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, 2000.

[4] 최윤석, "Middleware의 기본적인 이해 및 시장 분석," 한국오라클 Mobile Lab, 2005.05

[5] 한수, 박상현, 신승호, "U-Logistics환경 구축을 위한 다중 RFID 미들웨어 System의 설계 및 구현," 인천대학교 컴퓨터공학과, 2006.6

[6] Weilian Su, Ozgur B. Akan and Erdal Cayirci, Communication Protocol for Sensor Networks, Wireless Sensor Network, pp. 21-50, Kluwer Academic Publisher, 2004.

[7] Zigbee Device Object, "Zigbee document03525r5ZB," Zigbee Alliance, March 2004.

[8] Anna Hac, Wireless Sensor Network Designs, Wiley, 2004.

[9] Online: http://www.rfidjournal.com/article/articleview/858/1/82, Nicholas D. Evans, 'Middleware is the Key to RFID,' RFID Journal, 2004.4.5

[10] 김진태, 권영미, "RFID와 Zigbee를 이용한 유비쿼터스 u-Health System 구현," 전자공학회 논문지 제43권 TC편 제1호, pp.79-88, 2006.1