다양한 센싱 데이터 인식 트리거 컴포넌트 설계

The Component Design of a Diverse Sensing Data Recognition Trigger

김경옥, 반경진, 류남훈, 장문석, 김응곤
순천대학교 컴퓨터과학과

KyeongOg Kim, KyeongJin Ban, NamHoon Ryu,
MoonSuk Jang, EungKon Kim
Dept. of Computer Science,
Sunchon National University

요약

최근 들어 인간 중심의 정보화 사회가 USN 기술의 발전과 더불어 사물 간에도 정보들이 유기적으로 결합되고 활용될 수 있는 유비쿼터스 컴퓨팅 사회로 급격히 변모하고 있다. 이러한 유비쿼터스 컴퓨팅 사회를 성공적으로 구축하기 위해서는 센서노드들로부터 수집되는 센싱 정보를 효율적으로 관리하고, 센서 네트워크 구성에 대한 추상화 기능을 지원하는 USN 센서노드, 센서 네트워크, USN 미들웨어, 그리고 USN 응용 서비스 등의 USN 핵심 기술이 반드시 필요하게 된다. 본 연구에서는 각종 USN 단말장치들로 발생되는 센싱 값들의 전송 이벤트를 처리하기 위해서 USN 및 RFID 단말장치로부터 전송되는 다양한 센싱 값들을 효율적으로 처리할 수 있도록 데이터의 추상화를 수행하는 프로세스를 설계한다.

Abstract

Along with the advance of USN technology in a human-oriented informatization society these days, society is quickly changing into a ubiquitous computing society in which information even between objects can be organically combined and utilized. In order to successfully build such a ubiquitous computing society, it is indispensable to have core USN technologies such as USN sensor nodes, sensor networks, USN middleware, and USN applied services that efficiently manage sensing information collected from sensor nodes, and support the abstraction function for the composition of sensor networks. In order to process the transmission event of sensing values that are generated from various USN terminal devices, this study designs a process that performs the abstraction of data for the efficient process of diverse sensing values transferred from USN and RFID terminal devices.

Ⅰ. 서론

컴퓨터 및 네트워크 기술의 발전이 유비쿼터스 컴퓨팅 환경 구축을 가속화함에 따라 이를 활용하는 다양한 응용 서비스가 도입되었으며, 나아가 독립적으로 존재하는 응용 서비스들의 유기적 연결이나 통합에 대한 관심이 증폭되고 있다[1]. 특히, 인간 중심의 정보화 사회가 사람과 사물뿐만 아니라, 사물 간에도 정보들이 유기적으로 결합되고 활용될 수 있는 컴퓨터, 가전, 자동차, 생활기기, 의료기기, 교통시설 등의 모든 사물은 전자공간과 물리공간이 연계되는 USN(Ubiquitous Sensor Network)를 통해 연결된다. 이처럼, 유무선 통신기기와 센싱 기술의 발달로 인해, 인간의 접근이 어렵거나 지속적인 모니터링이 필요한 환경으로부터 센서를 이용하여 다양한 물리적인 현상을 수집하는 것이 가능해졌다[2].

현재 USN 환경에서 각 센서 노드들로부터 센싱 된 데이터를 기반으로 하는 응용 서비스는 상황 정보에 대한 이벤트를 감지하고 서버나 유저에게 이벤트의 발생을 알려서 즉각적인 대응을 유도하는 이벤트 알림 서비스가 주를 이룬다. 그러나 무한하고 연속적인 특성을 갖는 센싱 데이터에는 질의나 알림 서비스로는 알 수 없는 유용한 지식 정보가 존재할 수 있다. 그러므로 센서 데이터의 특

성을 고려한 연속적인 데이터 마이닝 기법을 적용하여 트레이닝 과정이나 패턴 추출과정을 통해 센서 데이터에서 유용한 지식 정보를 추출하고 이를 이용할 필요가 있다.

본 연구에서 각종 USN 단말장치들로 발생되는 센싱 값들의 전송 이벤트를 처리하기 위해서 각종 USN 및 RFID 단말장치로부터 전송되는 다양한 센싱 값들을 효율적으로 처리할 수 있도록 데이터의 추상화를 수행하는 프로세스를 Component based Development(CBD) 기반의 프레임워크를 설계했다. 본 논문의 구성은 다음과 같다. Ⅱ장에서는 센서 네트워크와 상황정보의 생성 및 관리에 대해 알아보고, Ⅲ장에서는 이벤트 인식 트리거 컴포넌트를 설계한다. Ⅳ장에서 Context의 특성 및 인식 Trigger에 대해 알아보고, Ⅴ장에서 결론을 통해 향후 연구 과제를 제시한다.

Ⅱ. 관련연구

1. 센서 네트워크

센서 네트워크는 수많은 센서노드들로 구성되어 있으며, 그 센서들은 원하는 데이터를 가장 잘 수집할 수 있는 최적의 위치에 배치되게 된다. 각각의 센서들은 감지능력, 간단한 처리 능력, 데이터 전송능력을 지니게 된다[3)4)]. 베이스 스테이션에서 쿼리를 보내고 이에 대한 응답을 받는 구조는 중앙집중식 상황인지에서 가장 많이 사용하는 방식이다. Directed Diffusion[5)]은 목표 이벤트에 대한 요구 사항을 'Interest'라는 패킷 형태로 네트워크에 플러딩하고 발생한 이벤트 정보를 베이스 스테이션으로 수집하는 방식이다. 베이스 스테이션으로의 전송 과정에서는 데이터 융합을 통해 정송되는 데이터의 양을 줄인다. TinyDB[6)]Cougar[7)]는 이러한 데이터 융합을 데이터베이스에 적용한 예이다. 하지만 이들 방식은 평균 온도를 얻는 것과 같은 동일 데이터 타입에는 데이터 융합을 통해 전송 데이터의 ㄷ양을 줄이지만 여러 타입의 이벤트로 이루어지는 복합 콘텍스트에 대해서는 연구가 이루어지지 않았다.

센서 네트워크내의 센서 노드들은 지능적, 능동적 상황 정보 생성이라는 측면에서 다음과 같은 문제점을 해결해야 한다. 첫째, 센서는 자립적 관리 능력을 포함해야 한다. 다양한 종류와 수많은 센서들을 위한 전력과 사용 중에 발생되는 오류 처리를 베이스 스테이션인, Sink 노드가 담당하는 것은 어려운 일이다. 따라서 센서가 갖는 컴퓨팅 기능을 사용한 센서들 간의 상호 관리 기능을 위한 기술이 필요하다. 둘 째, 정확도 높은 센싱 정보의 획득이다. 이를 위해서는 한 종류의 센서가 제공하는 정보뿐만 아니라 여러 종류의 센서들이 감지한 정보를 혼합하여 상호 보완적 센싱 정보를 생성하는 기능을 위한 센싱 정보 통합 모형이 중요한 역할을 담당한다.

2. 상황정보 생성 및 관리

최근에 소개되고 있는 USN 응용 서비스 모델은 센싱 정보를 획득하고 검증하여 단순히 사용자에게 제공하는 수준이 아니라, 수집된 센싱정보들을 이용하여 과거에 저장된 정보들과 비교 분석하고, 예측하고, 추론하여 새로운 상황정보를 생성할 수 있는 기능들을 필요로 하고 있다. 그러므로 USN 미들웨어는 상황정보 생성을 위하여 과거 수집된 정보 DB와 외부 비즈니스 DB 등을 연계하기 위한 기능과 상황정보 생성을 위한 규칙을 정의하고 이러한 규칙을 처리할 수 있는 방법 등을 지원해야 한다. 이러한 상황정보의 생성 기능은 센서노드 성능의 효율성과 다양한 USN 응용 서비스 모델을 지원하기 위하여 Server-side 미들웨어에서 뿐만 아니라, Innetwork 미들웨어에서도 지원 가능하도록 구현하는 것이 바람직하다[8)].

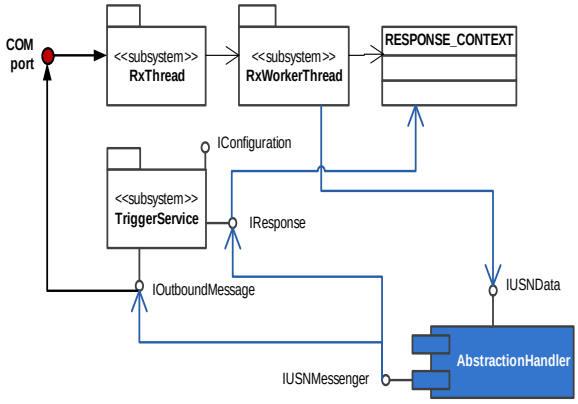
Ⅲ. 이벤트 인식 트리거 컴포넌트

각종 USN및 RFID 단말장치로부터 전송되는 다양한 센싱 값들을 효율적으로 처리하기 위해 데이터의 추상화를 수행하는 프로세서 CBD 기반의 프레임워크이다. Component 기반의 트리거 프레임워크는 크게 RS232C Port를 이용한 Serial 통신 처리부와 IOCP 기반의 Socket 토신 처리부 그리고 각 통신 처리부로부터 호출되는 데이터 추상화 Component로 구성된다.

1. COM Port Listener

COM Port Listener는 USN 단말장치와 Zigbee 무선

통신 기반의 RS-232C 통신을 수행하여 센싱 된 데이터를 수집하고, 필요한 경우 각 USN 단말장치와 Command를 송수신하는 기능을 수행한다. COMM Port Listener의 구조는 그림 1과 같다.



▶▶ 그림 1. COMM Port Listener의 구조

COM Port Listener는 COM 기반의 Windows Service 구조를 갖는 TiggerService를 통해서 초기에 자동으로 구동되고, 이때 지정된 COMM Port로부터 Packet Data를 수신하기 위한 RxThread와 수신된 Packet Data를 처리하는 RxWorkerThread가 Firing된다.

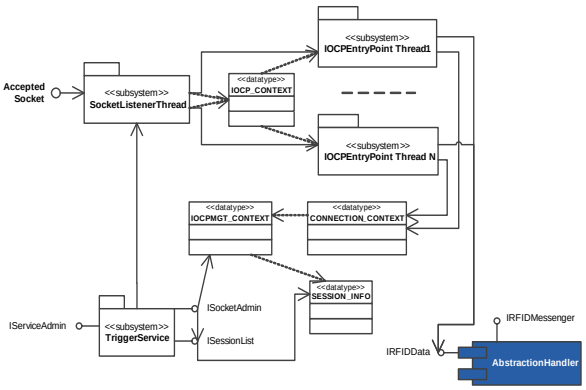
USN 단말장치로부터 송신된 Packet Data는 RxThread에 의해서 수신되고, 수신하는 과정에서 Packet의 Validation이 검증된다. 검증된 Packet Data는 그 형식에 따라 처리의 구분 실행을 위해서 RxWorkerThread로 수신된 Packet Data를 전달한다.

RxWorkerThread는 전달된 Packet Data의 형식에 따라 분리 처리를 수행하게 되는데, USN 단말장치로 요청된 Request에 대한 Response인 경우 이 Response Packet을 TriggerService의 IResponse Interface가 처리하도록 RESPONSE_PACKET에 저장한다. 이와 달리 만약 전달된 Packet Data가 EVENT인 경우 Packet Data의 실제 추상화 처리를 수행하는 AbstractionHandler::IUSNData Interface를 Packet Data와 함께 호출하게 된다.

2. IOCP Socket Listener

일반적으로 Overlapped I/O를 통한 Asynchronous I/O는 Synchronous I/O에 비해 기능적, 성능적으로 많은 장점을 갖는다. 그러나 Overlapped I/O는 접속된 Client 수만큼의 서버 프로그램 안에 Thread를 생성하면서 처리하는 구조를 갖기 때문에 잦은 Thread Switching에 의한 과부하와 Thread Resource 남용 등의 단점을 갖는다.

IOCP(Input Output Completion Port)는 이러한 단점을 극복하면서 Overlapped I/O를 통한 Asynchronous I/O의 장점을 수용한 I/O 모델로 Windows 계열의 운영체제에서 가장 빠른 네트워크 성능을 갖는다. 즉, IOCP는 Overlapped I/O와 이 I/O를 처리하는 Thread Pool을 조합해서 구성한 Thread Model을 사용한다. 따라서 본 시스템은 Socket 통신을 위해 그림 2와 같은 IOCP 기반의 비동기적 처리가 가능한 Socket Listener를 설계하였다.



▶▶ 그림 2. IOCP 기반의 Socket Listener 구조

TriggerService는 SocketListenerThread를 Firing하고 SocketListener-Thread는 다시 실제 처리를 담당하는 IOCPEntryPoint Thread를 Firing한다. 여기서 IOCPEntryPoint Thread는 Thread Pool에 의해서 관리되는 Thread들이다. SocketListenerThread가 Socket을 Accept하면 I/O가 Completion된 IOCP_CONTEXT 정보를 각 IOCPEntryPoint Thread에게 전달하고, 전달받은 IOCPEntryPoint Thread는 Socket Context 관리를 위해 CONNECTION_CONTEXT를 구성한 다음 이 값을 이용해서 IOCPMGT_CONTEXT에 다른 부가 정보들을 추가해서 이 값들을 저장한다.

3. Data 추상화 Component

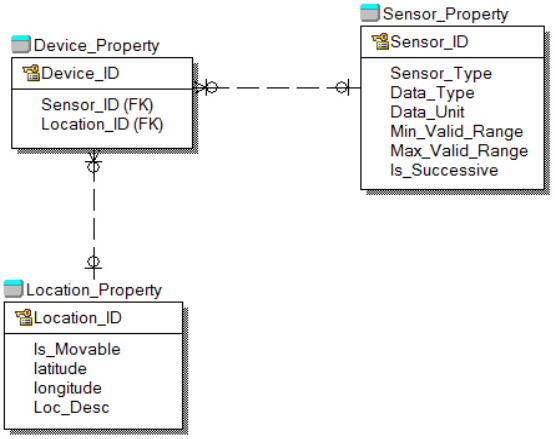
전송되는 USN 센싱 Data는 RFID Tag 정보와는 달리 미리 정의된 코드 값들과 측정된 센싱 값들의 연속으로 구성되기 때문에 그 형식이 매우 원시적인 구조를 갖는다. 따라서 이후 연속되는 프로세스들이 이러한 형식의 데이터 구조를 계속 사용하게 되는 경우, 연속 데이터들의 집합에서 특정 Index로 시작되는 값들을 분리해 내고 이를 해석하는 과정을 반복해서 원하는 값을 얻어 내게 된다. 표 1은 RFID/USN을 통해서 얻은 Tag 인식 Message이다.

표 1. Tag 인식 Message

```
<?xml version="1.0" encoding="utf-8"?>
<methodResponse><params><param>
  <value><struct>
    <member>
      <name>Location_ID</name>

<value><string>LOC_001</string></value>
    </member>
    <member>
      <name>Device_ID</name>

<value><string>USN_DEV_001</string></value>
    </member>
    <member>
      <name>Sensing_Value</name>

<value><string>251</string></value></member>
    <member>
      <name>Time_Stamp</name>

<value><string>20090306T15:18:57</string></val
ue>
    </member>
  </struct></value>
</param></params></methodResponse>
```

따라서 본 시스템은 Raw Data 형식의 USN Data를 초기 수집 이후의 여러 프로세스들이 추상화된 데이터를 다룰 수 있도록 함으로써 Data 처리에서 발생할 수 있는 오류를 제거하고자 Data 추상화 Component를 개발하였다. 즉, Data 추상화 Component는 센싱 된 Raw Data를 Database를 참조하거나 시스템 정보를 이용하여 다음과 같은 형식의 XML 형식의 추상화된 데이터를 생성하고, 이를 이후 프로세스들에게 전달하게 된다.

Data 추상화 Component는 추상화된 XML 데이터를 생성하기 위해서 필요한 경우 Database에 존재하는 장치, 센서, 위치 특성 정보들을 참조해 그림 3은 Database에 장치, 센서, 위치 특성 정보들을 저장하기 위한 Logical Scheme을 보이고 있다.



▶▶ 그림 3. 센서 정보 저장을 위한 Logical ERD

Sensor_Property Table은 센서가 갖는 일반적인 특성들을 저장하는데 다음과 같은 Attribute들을 포함한다.

- Sensor_Type: Analog/Digital
- Data_Type: 온도/습도/조도/거리/연기/…
- Data_Unit: ° C/%/lux/Cm/m/%/on_off/…
- Min_Valid_Range: 유효한 범위의 최소값
- Max_Valid_Range: 유효한 범위의 최대값
- Is_Successive: yes/no,
- 일정 시간 간격으로 연속적으로 측정되는 값인지 아닌지 여부

Location_Property table은 센서가 설치된 위치의 일반적인 특성들을 저장하는데 다음과 같은 Attribute들을 포함한다.

- Is_Movable: yes/no,
- 센서 노드의 위치가 움직일 수 있는 지 여부
- latitude: 위도 값(optional)
- longitude: 경도 값(optional)
- Loc_Desc: 위치 설명

Device_Property Table은 Attribute로 USN Node의 특성 Table의 ID와 위치 Table의 ID를 Foreign Key(FK)를 포함해서 필요한 정보를 참조하게 된다.

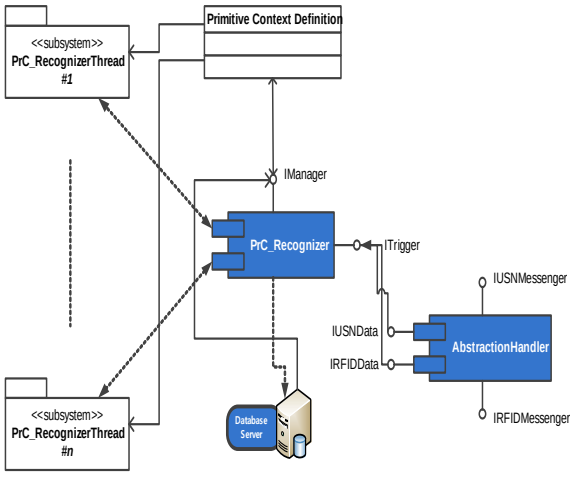
Ⅳ. Context의 특성 및 인식 Trigger

1. Primitive-Context의 특성 및 인식 Trigger 정의

XML 형식의 Data로 추상화된 센싱 데이터는 Primitive Context Recognizer(PrC_Recognizer) Component로 IAdmit Interface를 통해 전달된다. PrC_Recognizer Component는 전달된 데이터의 형식에 따라 이를 처리하는 Thread로 전송된 센싱 데이터를 Parameter로 전달하고, 전송 받은 Thread는 미리 정의된 Primitive Context 인식 조건을 시험해서 Primitive Context의 인식을 시도하게 된다.

이때 참조되는 Primitive Context Definition은 PrC_Recognizer Component가 처음 기동할 때 데이터베이스로부터 메모리에 Loading된다. 또한 이 정의 정보는 PrC_Recognizer Component의 IManager Interface를 통해 삽입, 삭제, 수정, Re-loading 등의 연산을 수행하게 된다.

Poly-Context의 인식 구조는 Primitive Context의 인식 구조와 유사하다. 그림 4에 나타난 바와 같이 PrC_Recognizer Component로부터 인식된 Primitive Context 정보는 Poly-Context Recognizer(PoC Recognizer)로 전달되고, 전달 받은 Primitive Context를 인식조건으로 갖는 Poly-context 인식 Thread로 전달하여 Poly-context의 인식을 시도한다.



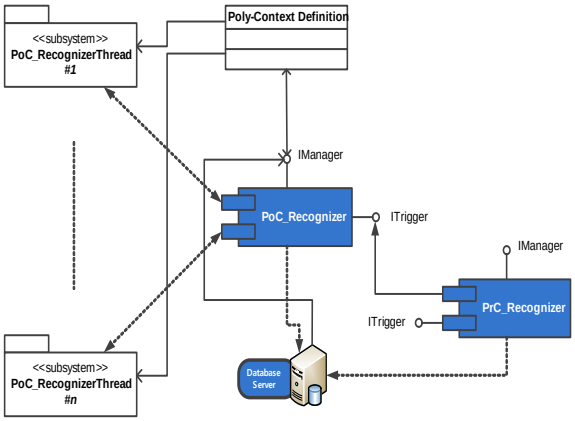
▶▶ 그림 4. Primitive Context Recognizer를 통한 인식 처리 흐름

2. Poly-Context의 특성 및 인식 Trigger 정의

Primitive Context의 경우와 유사하게 Poly-context의 인식을 위해 참조되는 Poly-context Definition은 PoC_Recognizer Component가 처음 기동할 때 데이터베이스로부터 메모리에 Loading된다. 또한 이 정의 정보는 PoC_Recognizer Component의 IManager Interface를 통해 삽입, 삭제, 수정, Re-loading 등의 연산을 수행하게 된다. 그림 5는 Poly-context Recognizer를 통한 인식 처리 흐름을 나타낸다.

Ⅴ. 결론

최근 빠른 속도로 유비쿼터스 컴퓨팅의 시대를 맞이하고 있다. 이러한 유비쿼터스 컴퓨팅은 언제, 어디서나, 어떠한 정보라도 편리하게 전송 및 처리가 가능하다는 점에서 미래의 핵심 기반 기술로 자리매김해 나가고 있다.



▶▶ 그림 5. Poly-context Recognizer를 통한 인식 처리 흐름

본 연구에서는 각종 USN 단말장치들로 발생되는 센싱 값들의 전송 이벤트를 처리하기 위해서 각종 USN 및 RFID 단말장치로부터 전송되는 다양한 센싱 값들을 효율적으로 처리할 수 있도록 하였으며, 향후 과제로는 유비쿼터스 환경에서 발생 가능한 일시성, 연속성, 이벤트 질의와 같이 다양한 형태의 질의도 처리할 수 있는 연구가 진행되어야 할 것이다.

감사의 글

본 연구는 교육과학기술부와 한국산업기술재단의 지역혁신 인력양성사업으로 수행된 연구결과임

■ 참 고 문 헌 ■

[1] 최황영, 김재혁, 서오석, "RFID/USN 환경에서의 이기종 서비스 시스템간 시나리오기반 협업지원 미들웨어 플랫폼 개발", 한국 RFID/USN 협회, 제4회 RFID/USN 연구논문, 2008.

[2] 김민수, 이용준, 박종현, "USN 미들웨어 기술개발 동향", 전자통신동향분석, 제22권, 제3호, 2007.

[3] I.F. Akyildiz, "A Survey on Sensor Network", IEEE Communication Magazine, pp.102-114, August 2002.

[4] Philippe Bonnet, Johannes Gehrke, Praveen Seshadri, *Towards Sensor Database Systems*, Springer Berlin, 2001.

[5] C. Intanagonwiwat, R. , Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks", In ACM MOBICOM., pp.56-67, 2000.

[6] S.Madden, "TinyDB: An Acquisitional Query Processing System for Sensor Networks", ACM Transaction on Database Systems, Vol.30, No.1, pp.122-173, 2005.

[7] Philippe Bonnet, Johannes Gehrke, Praveen Seshadri, "Towards Sensor Database Systems," Springer-Verlag, pp.3-14, 2001.