
USN 미들웨어 설계사례를 통한 패턴지향 아키텍처 설계방법의 개선

Improvement of Pattern Oriented Software Architecture Design Approach with Empirical Design of USN Middleware

공상환
백석대학교 정보통신학부

Sang Hwan Kung(kung@bu.ac.kr)

요약

센서 네트워크는 무선통신기술의 발전에 힘입어 다양한 응용분야에서 센서들을 이용함으로써 무인화와 자동화를 가능하게 하고 있다. 센서 네트워크를 가능하게 하는 기술 중의 하나는 센서로부터 얻어지는 정보들을 효율적으로 수집하고 분석해서 자동적으로 제어하는 역할을 담당하는 USN(Ubiquitous Sensor Network) 미들웨어 기술을 꼽을 수 있다. 논문에서는 버섯재배 시스템의 구현을 사례로 하여, 버섯재배 응용을 위한 USN 미들웨어 설계를 수행하는 과정에서 요구되는 효율적인 소프트웨어 아키텍처 개발방법론을 다루고 있다. 특히, 아키텍처 설계방법 중 아키텍처 패턴을 효율적으로 이용하는 방법을 제안하고, 사례 시스템의 설계에 개선된 방법을 적용해 봄으로써 타당성을 검증하고자 하고 있다. 아울러, 기존의 소프트웨어 아키텍처 연구에서 간과되고 있는 소프트웨어 아키텍처를 위한 품질속성들을 검토하고, 정리하여 함께 제시하고 있다.

■ 중심어 : | 소프트웨어 아키텍처 | 아키텍처 패턴 | 소프트웨어 품질 | 유비쿼터스 센서 네트워크 | USN 미들웨어 | USN 응용 |

Abstract

The Sensor Network enables many distributed systems to be unmanned and automated by using of diverse sensors as well as wireless communication technologies. One of major enabling technologies for the sensor network is the USN middleware which plays the role of collecting and analyzing of measurements of sensors and controlling of the environments. The paper deals with the fungus cultivating environment based on Sensor Networks. Especially, we focus on the design of USN middleware for the embedded system, and explain how to design software architecture in terms of architectural patterns. In this design process, the improvement of methodology for pattern-oriented architecture design is proposed and the quality attributes for the architecture design is newly classified and suggested for the reference of software architecture design.

■ keyword : | Software Architecture | Architecture Pattern | Software Quality | Ubiquitous Sensor Network | USN Middleware | USN Application |

I. 서론

농작물의 재배를 위한 자동화 시설은 영농방법을 개

선하기 위한 오랜 동안의 과제였다. 이를 위해 기계 중심적인 방식의 도구나 장치들이 많이 개발되어 영농현

접수번호 : #070814-003

접수일자 : 2007년 08월 14일

심사완료일 : 2007년 10월 08일

교신저자 : 공상환, e-mail : kung@bu.ac.kr

장에서 활용되어 왔다. 이러한 발전은 최근 들어 USN 이나 RFID(Radio Frequency IDentification) 등을 중심으로 한 유비쿼티스 기술이 출현함에 따라 다시 기계화로부터 전자화, 무선화로의 전환이 가능하게 되었다.

최근 들어서는 무선통신이나 하드웨어 및 배터리의 소형화 등의 연구에 힘입어 USN 기반의 다양한 응용들이 활성화되었는데, 특별히 상용화가 이루어지고 있는 분야로는 환경감시, 원격재배, 환자상태감시, 교각강도 감시, 삼림감시, 화재감시 등이 있다[1][2].

본 연구는 이러한 응용들 중의 하나인 벚꽃재배를 위한 USN 환경을 구축하기 위해 요구되는 센서 네트워크 시스템의 설계를 다룬다. 특히, 연구의 초점을 임베디드 기반의 USN 미들웨어 설계에 초점을 맞추어, 잘 알려진 아키텍처 패턴들을 이용하여 어떻게 소프트웨어 아키텍처를 실현할 것인가를 설명하고 있다.

II. 관련연구

2.1 USN과 미들웨어 연구동향

USN 미들웨어는 물리적으로 응용서비스와 센서 네트워크 인프라의 중간에 위치한다. 응용 서비스를 효율적으로 지원하기 위하여 센서 노드와 게이트웨이, 그리고 서버 노드에 분산하여 위치하고, 상호 연동을 통해 서비스를 제공한다.

그동안 응용이나 센서 네트워크의 특성에 기반하여 다양한 미들웨어가 개발되어 활용되어 왔다[3]. 이러한 미들웨어는 구조설계에서 어떠한 관점에 관심을 두고 있는냐에 따라 다양한 유형들로 구분되고 있다. 하나의 분류방식은 데이터 기반(Data Centric)구조, 이벤트 기반(Event Centric) 구조, 이동 에이전트(Mobile Agent) 기반의 구조, 제한자원 보호기반의 구조, 공통 센서(Common Sensor) 기능 기반의 구조, QoS 지원 구조, 그룹 접근(Group Approach) 구조로 구분하는 것이다 [4].

2.2 패턴기반 아키텍처 개발방법

아키텍처 패턴은 소프트웨어 시스템의 기본 골격인

소프트웨어 아키텍처를 구축할 때 적용하는 패턴이다. 이러한 패턴은 통상 시스템의 구성요소와 이 구성요소 사이의 구조적인 관계나 동적인 상호작용, 그리고 제약 조건으로 정의된다[5]. 이 패턴은 앞서 설명한 품질속성이나 품질속성 달성전략과 깊은 관련이 있다고 할 수 있는데, 그것은 아키텍처 패턴 역시 품질속성을 달성하기 위한 품질속성 달성전략을 보다 정형화하고 구조화 한 결과이기 때문이다.

아키텍처를 설계하는 과정에서는 참조되는 품질속성은 기능성과 신뢰성, 사용용이성, 유지보수성, 이식성, 효율성을 중심으로 ISO-9126에 제시되어 있다. 그러나 이 속성들은 전반적인 소프트웨어의 품질속성으로 아키텍처에 적용하기에는 미흡한 점이 많다.

아키텍처 설계는 이러한 패턴과 품질속성을 기반으로 수행한다. 이러한 아키텍처 설계를 위해 연구된 방법론은 다양하게 제시되고 있으나, 중요한 절차나 흐름에 있어서는 방법론들 사이에 커다란 차이가 없다. 즉, 시스템이 제공해야 하는 기능적인 요구사항을 충족하기 위해 설계요소의 대상이 되는 요구기능들을 단계적으로 분해해 가면서 설계를 단계적으로 수행한다. 그러나 주목할 사항은 이러한 설계요소를 분해해 나가는 과정에서 미리 정의되어 있는 아키텍처 패턴을 이용한다는 점이다[6][7].

아키텍처 설계의 각 단계에서 수행되는 중요한 활동을 살펴보면 다음과 같다.

- 설계가 착수되면 제일 먼저 시스템의 배경도에 해당되는 비즈니스 컨텍스트를 작성한다.
- 아키텍처를 위한 핵심적인 요구사항을 기능 요구사항과 시스템의 품질에 해당되는 비기능 요구사항으로 구분하여 작성한다.
- 설계할 모듈과 그 모듈을 설계하는 데 필요한 품질속성을 선택한다.
- 모듈의 설계에 참조가 되는 아키텍처 패턴을 참조한다.
- 참조패턴을 이용하여 설계요소에 대한 설계를 한다.
- 더 이상의 분해가 필요 없을 때까지 설계요소를 분해하고 설계하는 절차를 반복한다.

III. USN 미들웨어 사례 분석

3.1 USN 기반의 버섯재배 응용

[그림 1]은 USN 기반의 버섯재배 시스템의 구성을 보여주고 있다. 각 방에는 환경의 감지를 위한 온도센서나 습도센서, 연기센서, 도어센서가 각각의 ID와 함께 설치된다. 이들은 여러 계층의 셸프에 설치되기 때문에, 유형별로 복수 개의 센서가 다른 장소에 설치되기도 한다. 센서 외에도, 센서에 의해 감지된 정보를 토대로 운영되어야 하는 에어컨, 통풍기, 난방기도 각 방에 함께 설치된다. 이러한 센서나 장치들은 각 방의 바로 외부에 설치되는 제어기에 의해 통제된다.

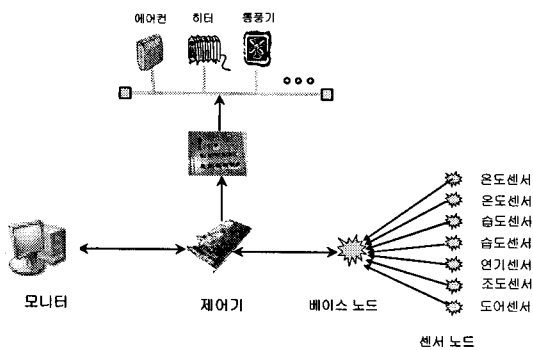


그림 1. USN 기반 버섯재배 시스템의 구성

3.2 아키텍처 요구사항 분석

버섯재배 응용은 하드웨어와 소프트웨어가 통합된 시스템이다. 특히 소프트웨어가 설치될 노드도 USN 노드 뿐 아니라, 제어기의 역할을 수행할 임베디드 시스템과 PC가 있다. 논문에서는 USN 기반의 버섯재배 시스템 중 소프트웨어의 아키텍처가 가장 중요한 임베디드 기반의 제어기의 소프트웨어 설계에 초점을 맞추어 설계한다.

임베디드 기반의 제어기 소프트웨어가 고려해야할 기능적인 요구사항을 살펴보면 다음과 같다.

- 시스템은 각각의 USN 노드들의 설치현황을 유지하여야 하며, 변경할 수 있어야 한다.
- USN 노드로의 폴링(polling), 스케줄링, 이벤트, 제어 등에 의한 처리가 가능하여야 한다.

- USN 노드로부터 전달되는 모든 메시지는 발생 즉시 수집되어야 한다.
- 수집된 메시지는 분석을 위해 메시지의 유형과 센서의 유형별로 분류 및 저장되어야 한다.
- 운용자가 지시한 명령어와 응답은 로깅되어야 한다.
- USN노드의 지속적 변화상태는 감시되어야 한다.

IV. 패턴 기반 소프트웨어 아키텍처의 설계

4.1 소프트웨어 아키텍처의 설계

예제로 설명할 소프트웨어 아키텍처 설계의 주요 범위는 버섯재배 응용의 핵심부분이라고 할 수 있는 임베디드 보드의 제어기 소프트웨어로 한다. 이 제어기의 아키텍처 설계를 위해서는 요구된 기능을 설계모듈로 전환하는 것이 필요한데, 이때 관련된 패턴과 품질속성이 고려가 된다. 본 연구에서는 패턴과 품질속성을 연계시키는 과정을 기능을 위한 패턴을 선정하는 과정과 선정된 패턴들을 다시 배치하는 과정으로 구분하여 단계적인 정제를 수행하고 있다.

4.2 기능요소별 아키텍처 패턴의 선별

패턴 기반의 아키텍처 설계에서 특정한 설계대상 요소가 선정되면, 먼저 설계요소의 분해에 필요한 패턴을 선택하게 된다. 이때 적절한 패턴의 결정은 설계요소가 어떠한 기능 및 품질요구사항을 갖느냐에 의해 결정된다. 여기서는 요구사항에서 정의된 기능들을 토대로 패턴을 선택하는 절차를 설명하기로 한다.

(1) USN 노드 명령어 처리를 위한 패턴

- 패턴의 대안
 - Thread for Message(메시지 당 쓰레드) 패턴 : 하나의 명령어 입력에 대해 하나의 쓰레드를 할당하여 처리하는 패턴이다.
 - Object for Message(메시지 당 객체) 패턴 : 하나의 명령어 입력에 대해 하나의 객체를 할당하는 패턴이다.

- 설계 결정
제어기의 운영환경은 입출력 환경의 제약으로 인해 멀티태스킹에 적합한 Thread for Message 패턴 보다는 Object for Message 패턴이 적합하다.
- 고려된 QoS : 자원경제성

(2) USN 노드와의 통신을 위한 패턴

- 패턴의 대안
 - Asynchronous Channel(비동기 채널) 패턴 : 두 개의 노드 사이에서 단방향 통신만을 제공하는 패턴이다.
 - Synchronous Channel(동기식 채널) 패턴 : 데이터를 송신을 하는 동안에도 수신 가능한 패턴이다.
- 설계 결정
USN 응용에서 데이터 통신의 유형은 USN 노드의 제어명령과 같이 질의 및 응답형태의 양방향 통신 뿐 아니라, USN 노드로부터의 정보 메시지의 출력과 같은 단방향 통신도 가능하다. 따라서 통신 요구사항은 USN 응용 내의 작업유형에 따라 Asynchronous Channel과 Synchronous Channel이 동시에 필요하다.
- 고려된 QoS : 자원공유성

(3) USN 노드의 자료수집을 위한 패턴

- 패턴의 대안
 - Communicating Process(통신하는 프로세스) 패턴 : 여러 개의 병렬처리 프로세스나 또는 쓰레드가 실행되는 패턴이다.
- 설계 결정
USN 응용에는 언제 입력이 발생 할지 모르는 복수의 USN 노드가 연결되어 있어, 항상 입력을 대기하는 독립적인 쓰레드가 필요하다.
- 고려된 QoS : 병렬성

(4) 수집된 자료의 통합을 위한 패턴

- 패턴의 대안
 - Producer-Consumer(생산자-소비자) 패턴 : 이

패턴은 하나의 버퍼를 공유하는 불특정 생산자와 복수의 수신자로 구성된다.

- 설계 결정
USN 노드로부터 수집된 자료를 하나의 버퍼에 넣는 쓰레드는 제공자의 역할을 담당하고, 이 버퍼에서 저장된 자료를 읽는 객체는 소비자의 역할을 담당한다. 이 패턴은 다대일의 관계로 복수의 채널로부터 전달되는 정보를 하나의 모듈로 전달하는 multiplexing 서비스를 위하여 활용된다.
- 고려된 QoS : 자원공유성

(5) 자료의 분배를 위한 패턴

- 패턴의 대안
 - Publisher-Subscriber(출판자-구독자) 패턴 : 복수의 구독자는 관심이 있는 데이터의 ID를 먼저 등록한다. 출판자가 ID와 함께 메시지를 출판하면, 이 메시지는 미리 신청한 구독자들에게 분배된다.
 - Distributer Pattern(분산자) 패턴 : 이 패턴은 메시지를 복수의 수신자에게 전달할 때 미리 고정된 수신자를 정해 놓고 지정된 분배자에게만 자료를 분배하는 패턴이다.
- 설계 결정
USN 노드로부터 얻어진 메시지는 유형이 다양하며, 또한 메시지의 수신자도 다양하다. 따라서 모듈의 유형이나 수에 종속되지 않고 메시지를 분배하기 위해서는 Publisher-Subscriber 패턴이 적합하다.
- 고려된 QoS : 자원공유성

(6) 구성관리를 위한 패턴

- 패턴의 대안
 - Blackboard(블랙보드) 패턴 : 정보를 공유하는 복수 개의 정보소스들이 존재한다. 이 중 일부는 정보를 갱신하며, 또 다른 일부는 이 정보를 활용한다.
- 설계 결정
USN 노드들의 설치 및 상태에 대한 정보를 집중적

시지는 메시지의 이용자에게 각각 Full mesh 형태로 분배하기 보다는 하나의 메시지 수집기를 두어 메시지를 재분배 하는 것이 구조를 단순화하고 유연성도 보장된다.

- 고려된 QoS : 정보흐름의 단순성

(3) 통신 관리기의 배치

- 설계 검토
각각의 통신채널 처리기를 종합적으로 관리해 주는 별도의 모듈이 필요하며, 임의의 응용은 특정한 채널과 통신을 하기 전에는 이에 앞서 통신채널 관리기로부터 채널의 주소를 확인한다.

- 고려된 QoS : 접근용이성

(4) 메시지 저장기의 배치

- 설계 검토
USN과의 통신에서 획득한 메시지는 저장되어야 한다. 한편, 이 메시지는 크게 명령에 의한 응답 메시지와 측정 또는 경보 메시지 처럼 다양하다. 따라서 메시지 저장을 담당하는 별도의 모듈이 필요하며, 이 모듈은 메시지의 유형에 따라 저장 및 접근을 담당할 별도의 모듈을 운영하게 된다.

- 고려된 QoS : 자원관리의 용이성

(5) 메시지 필터의 배치

- 설계 검토
상황을 관리하기 위해서 모든 메시지가 필요한 것은 아니며, 상황에 변화를 표현하는 메시지의 입력만 있으면 된다. 이를 위해 메시지 필터의 배치가 요구된다.

- 고려된 QoS : 자원경제성

(6) 프로세스 관리자의 배치

- 설계 검토
시스템 내에는 복수의 쓰레드들이 실시간으로 수행된다. 따라서 이들을 먼저 기동해 주어야 할 프로세스 관리자가 필요하다. 운영자의 기동명령에 의해 프로세스 관리자가 동작을 하면, 이 프로세스

관리자는 시스템에서 실시간으로 동작해야 할 실시간 쓰레드들을 기동시킨다.

- 고려된 QoS : 동적제어성, 접근용이성

V. 평가

본 연구의 아키텍처 설계과정에서 성과라고 하면, 크게 두 가지를 들 수 있는데, 하나는 아키텍처 개발방법에서의 절차적인 개선을 이루었다는 점이고, 다른 하나는 아키텍처 설계에서 고려해야 할 품질속성을 구체화하였다는 점이다.

본 연구에서 제시하고 있는 Pattern Oriented Software Architecture Design Approach(POSADA)를 소프트웨어 아키텍처 설계방법으로 잘 알려진 Attribute Driven Design(ADD) 방법[8]와 Architecture Based Design Method(ABDM) 방법[6]과 몇 가지 관점에서 비교를 해보면 다음의 [표 1]와 같다.

표 1. 아키텍처 설계방법론의 비교

비교 관점	ADD	ABDM	POSADA
방법론의 특징	품질속성 기반의 아키텍처 설계 방법론	설계 뷰와 패턴 기반의 아키텍처 설계 방법	설계 뷰와 패턴 기반의 아키텍처 설계 방법
설계 프로세스	분해모듈 선택 분해모듈의 분해·정제 - 아키텍처 동인 선택 - 아키텍처 패턴의 선택 - 기능의 할당 - 하위 I/F 결정 - 설계 검증 모듈의 분해설계 반복	개념적 설계 - 논리 뷰 : 스타 일을 고려한 설계 - 동시성 뷰 - 배치 뷰 구체적 설계	분해 모듈 선택 설계 뷰의 결정 분해를 위한 패턴 선택 패턴기반 설계 설계요소의 배치 설계요소간의 조정 뷰 간 동기화 검토 모듈 설계의 반복
복수의 설계관점에 대한 뷰의 고려	설계 뷰에 대한 고려가 없음	설계 뷰를 고려함	설계 뷰를 고려함
뷰 간의 동기화	전혀 고려하지 않음	구체적인 명시 없음	뷰간의 동기화를 통한 조정과정을 도입함
품질속성의 고려	일반적인 소프트웨어 품질속성을 고려함	일반적인 소프트웨어 품질속성을 고려함	아키텍처를 위한 품질속성을 고려함

연구결과 중 아키텍처를 위한 품질속성의 분류는 다음의 [표 2]에 기술되어 있다. 제시된 품질모델은 ISO-9126 품질모델이 소프트웨어의 상위 레벨이나 하위레벨을 포괄적으로 포함하는 품질을 개선하여 아키텍처 레벨의 품질을 위하여 필요한 품질특성들을 구체화하고 조정한 결과이다.

표 2. 아키텍처 품질속성

품질 특성	품질 부특성
가능성	완전성, 정확성
보안성	모듈 보안성, 통신 보안성
신뢰성	무결성, 가용성, 오류허용성
사용용이성	이해성, 학습성, 조작성, 대화성
유지보수성	수정용이성, 확장성, 시험용이성
이식성	하드웨어 독립성, 소프트웨어 독립성, 설치용이성, 재사용성
성능	시간경제성, 자원경제성, 병렬성, 동시성
공유성	모듈 공유성, 자원공유성
통제용이성	동적제어성, 구성제어 용이성, 자원관리 용이성
접근성	접근 용이성, 정보흐름의 단순성, 정보흐름경로의 거리
유연성	규모성, 호환성, 상호운영성
구현용이성	상세설계용이성, 프로그래밍용이성, 패키징용이성

VI. 결론

논문에서는 버섯재배의 자동화에 필요한 USN 기반의 시스템 특히, 소프트웨어의 아키텍처에 대한 설계를 다루었다. 센서 네트워크를 활용한 버섯재배 환경을 구현하는 과정에서 패턴기반의 아키텍처 설계방법론을 활용하고 있다. 분산환경에서의 하드웨어 구성요소와 각 구성요소의 역할 분담을 토대로 하여, 논문의 가장 핵심부분이라고 할 수 있는 제어기의 미들웨어에 대한 설계를 다루었다. 논문의 초점은 이러한 미들웨어를 설계하는 과정에서 아키텍처 패턴 지향의 설계방법론을 활용한다는 점이다. 패턴을 활용하기 위해서는 패턴들과 품질속성을 연계시키는 과정이 필요한 데, 논문에서는 이 과정을 설계요소를 위한 패턴의 선택과 아울러 패턴의 조정 및 배치를 위한 품질속성을 고려를 통해 보다 체계적인 설계를 수행하도록 하였다. 또한, 논문에서는 이러한 패턴들을 설계요소에 적용하고, 또 배치에 적용하면서 고려가 되었던 품질속성들을 토대로 소프

트웨어 아키텍처를 위한 품질속성을 분류하고 정의하여 보았다. 본 연구와 관련하여 아키텍처 패턴들을 선택하고 조정하는 운영방법(operation method)를 정형화 하는 연구는 향후의 과제로 남아 있다.

참고 문헌

- [1] A. Mainwaring, J. Polastre, R. Szewczyk, and D. Culler, "Wireless Sensor Networks for Habit Monitoring," ACM Sensor Networks and Applications, Sep. 2002.
- [2] N. Xu, et. al., "A Wireless Sensor Network for Structural Monitoring," ACM Sensys, Nov. 2004.
- [3] 김민수, 김광수, 이용준, "USN 마들웨어의 특징 및 기술개발 동향", 정보통신연구진흥원 주간기술동향, 제1284호, 2007(2).
- [4] 김영만, "USN 인프라 서비스 지원을 위한 계층적 미들웨어 플랫폼 구조", 정보처리학회지 특집 : RFID/USN, 제12권, 제5호, 2005(9).
- [5] M. Klein and R. Kazman, "Attribute-Based Architecture Style," Technical Report CMU/SEI-99-TR-022, CMU Software Engineering Institute, 1999.
- [6] F. Bachmann, L. Bass, G. Chastek, P. Donohoe, and F. Perzzi, "Architecture Based Design Method," Technical Report CMU/SEI-2000-TR-001, CMU Software Engineering Institute, 2000.
- [7] L. Bass and R. Kazman, "Architecture-Based Development," CMU Software Engineering Institute, Technical Report CMU/SEI-99-TR-007, ESC-TR-99-007, 1999.
- [8] R. Wojcik and et al, "Attribute-Driven Design(ADD)," Version 2.0, Technical Report CMU/SEI-2006-TR-023, CMU Software-Engineering Institute, 2006.

저자 소개

궁 상 환(Sang-Hwan Kung)

정회원



- 1977년 : 숭실대학교 전자계산학과 졸업(이학사)
- 1983년 : 고려대학교 대학원 전자정보처리학과 졸업(경영학석사)
- 1998년 : 충북대학교 대학원 전자계산학과 졸업(이학박사)
- 1981년 ~ 1998년 : 한국전자통신연구원 책임연구원
- 20001년 ~ 현재 : 백석대학교 정보통신학과 부교수
<관심분야> : 소프트웨어 구조, 분산시스템