프로덕트 라인 기반의 센서 네트워크 응용 소프트웨어 개발

김 영 희[†] · 이 우 진[††] · 최 일 우[†††]

요 약

현재 센서 네트워크 응용 분야는 소프트웨어의 효율적 개발을 위한 다양한 연구가 진행되고 있다. 이러한 연구들은 대부분 단일 센서 네트워크 응용 소프트웨어를 쉽고, 빠르게 개발하기 위한 방법에 중점을 두고 있다. 그러나 센서 네트워크 응용 소프트웨어는 운영체제의 핵심 모듈을 기반으로 다양한 종류의 센서 노드들을 제어하는 특징을 가지므로, 다양한 센서 네트워크 어플리케이션의 핵심 공통 기능을 정의하고 특정 센서 네트워크 어플리케이션의 워크플로우에 따라 가변적인 센서 노드들의 다양한 기능을 취사 선택하여 개발하는 방법이 효율적이다. 본 논문에서는 이러한 특성에 따라 소프트웨어 프로덕트 라인 기법을 센서 네트워크 응용 소프트웨어 개발에 적용, 센서 네트워크 응용 소프트웨어 도메인의 공통성을 식별하고 어플리케이션의 가변성에 따라 선택적인 개발을 지원하는 개발 사례를 제시하였다. 제시하는 사례를 통하여 일반적인 비즈니스 도메인과 비교하여 센서 네트워크 응용 도메인의 프로덕트 라인 구축을 위한 특성을 살펴보고, 제시한 가변성 피처 모델과 VEADL을 통하여 설계한 센서 네트워크 응용 도메인의 핵심 자산이 각 노드의 기능에 따라 선택적으로 재사용되는 적용 효율성을 보인다.

키워드 : 센서 네트워크, 응용 소프트웨어, 프로덕트 라인

Development of Ubiquitous Sensor Network Applications based on Software Product Line Approach

Younghee Kim[†] · Woojin Lee[††] · Ilwoo Choi[†††]

ABSTRACT

Currently various techniques for efficiently developing sensor network applications are developed. However, these techniques provide the method for developing only single sensor network application easily and rapidly. Since sensor network applications control various sensor nodes based on core components of operating system, the technique to develop applications by defining common functionalities of various applications and selecting variable functionalities according to the work flow of specific application is efficient. Accordingly, this paper presents an experimental study that identifies commonality of sensor network application domain and supports optional development according to the variability of application by applying product line approach to developing sensor network application. Through the experimental study, we describe the characteristics of sensor network application domain compared with general business domain for product line development. Also, we show the effectiveness of the proposed approach by presenting that core assets designed using the proposed variability feature model and VEADL are reused according to the functionalities of each sensor node.

Key Words : Sensor Network, Application Software, Product Line

1. 서 론

유비쿼터스 센서 네트워크[1,2]는 현재 다양한 종류의 센서 디바이스를 활용하여 여러 응용분야에 적극 활용 중이다. 센서 네트워크는 다양한 역할을 수행하는 센서 노드들로 구성이 되므로, 센서 네트워크 응용 소프트웨어는 수많은 센서 디바이스에 대한 다양한 기능을 수행하도록 구성하여야 한다.

센서 네트워크를 구성하는 노드 간의 신뢰도, 한정된 자원 등을 고려하여 응용 소프트웨어를 개발하는 것은 매우 복잡한 작업이다. 센서 네트워크를 구성하는 노드들은 크게 센서 노드, 라우터 노드, 싱크 노드, 엑츄에이터 노드로 역할을 구별할 수 있는데, 역할에 따라 응용 소프트웨어의 구성은 달라진다. 왜냐하면 센서 네트워크를 구성하는 노드들은 자원이 한정되어 있으므로, 불필요한 모듈을 노드에 탑재하지 않고 핵심 모듈만을 선택적으로 탑재하는 것이 효율적이기 때문이다. 또한 센서 네트워크 응용 소프트웨어는 특성상 수많은 센서 디바이스들의 다양한 역할 수행을 위한 가변적인 부분과 센서 네트워크 응용 소프트웨어 개발을 위한 핵심 모듈의 공통적 부분을 구분하여 접근하는 것이 효율적이다. 이러한 다양한 디바이스와 기능에 따라 복잡한 센서 네트워크 응용 소프트웨어를 효율적으로 작성할 수 있도록 하는 기술이 요구되고 있다.

† 정 회 원 : 숭실대학교 전산원 소프트웨어정보학과 교수
†† 정 회 원 : 한국정보통신대학교 공학부 Postdoc.
††† 정 회 원 : 강남대학교 교양교수부 교수
논문접수 : 2007년 9월 12일, 심사완료 : 2007년 11월 27일

프로덕트 라인[3] 기반 개발 기법은 이러한 센서 네트워크 응용 분야의 특성을 만족하는 기법 중 하나이다. 핵심 모듈의 공통적인 부분을 핵심 자산(Core Asset)으로 구축하고 다양한 디바이스와 기능에 따라 선택적으로 재사용 하여 복잡한 센서 네트워크 응용 소프트웨어의 개발을 효율적으로 지원할 수 있기 때문이다.

본 논문에서는 프로덕트 라인 기반 개발 기법을 센서 네트워크 응용 도메인에 적용한 적용 사례를 제시한다. 사례는 센서 네트워크 응용 도메인의 특성에 알맞게 프로덕트 라인 개발 기법을 커스터마이징하여 적용하였다. 도메인 공학에서 센서 네트워크 응용 도메인의 스코핑을 효율적으로 수행하기 위한 피처-프로덕트 매핑 테이블(Feature-to-Product Mapping Table)과 센서 네트워크 응용 소프트웨어의 가변적 피처의 다양성을 표기하기 위한 UML기반의 가변성 피처 모델(Variability Feature Model)을 제시하였고, 아키텍처 관점에서 선택적으로 재사용되는 가변성을 기존의 ADL(Architecture Description Language)을 확장하여 VEADL(Variability Extended ADL)을 제시하였다. 이를 통하여 센서 네트워크 도메인 공학의 결과들이 선택적으로 재사용되어 응용 소프트웨어의 효율적 개발이 가능하다.

2. 관련 연구

이 장에서는 프로덕트 라인 개발 기법에 대하여 소개하고, 센서 네트워크 응용 소프트웨어를 개발하는데 있어서 프로덕트 라인 개발 기법의 적용 필요성에 대하여 설명한다. 또한 기존의 센서 네트워크 응용 소프트웨어를 개발하기 위한 여러 기법들에 대하여 설명한다.

2.1. 프로덕트 라인 개발 기법

프로덕트 라인 개발 기법은 한 프로덕트 라인에 속하는 여러 어플리케이션들이 공유할 수 있는 핵심 자산을 재사용하는 대표적인 소프트웨어 재사용 방법[3]으로 현재 FOPLE(Feature Oriented Product Line Engineering)[4], PuLSE(Product Line Software Engineering)[5]등, 다양한 연구와 적용이 이루어지고 있다. 프로덕트 라인 개발 기법은 유사 도메인의 기능적 공통성과 가변성에 기반하여 공통 기능을 포함하는 핵심 자산의 개발과 이를 이용한 프로덕트 개발의 측면을 포함하고 있다. 핵심 자산은 프로덕트 라인의 여러 멤버에서 재사용될 수 있기 때문에, 공통성과 가변성의 명확한 정의는 높은 재사용성을 제공, 생산성을 향상시켜 고품질의 어플리케이션을 빠른 시간 내에 개발하는데 필수 요소이다.

프로덕트 라인 개발 기법은 일반적으로 크게 도메인 공학과 응용 공학으로 나뉜다. 도메인 공학에서는 다양한 응용 소프트웨어를 위한 공통피처를 식별하고, 핵심 자산을 구축하고, 응용 공학에서는 도메인 공학의 결과들을 선택적으로 재사용하여 특정 소프트웨어를 개발한다.

센서 네트워크는 다양한 역할을 수행하는 센서 노드들로 구성이 되므로, 센서 네트워크 응용 소프트웨어는 수많은 센서 디바이스에 대한 다양한 기능을 수행하도록 구성하여야 한다. 그러나 센서 네트워크를 구성하는 노드들은 자원이 한정되어 있으므로, 불필요한 모듈을 노드에 탑재하지 않고 핵심 모듈만을 선택적으로 탑재해야 최적의 네트워크를 수행할 수 있다. 이러한 특성으로 인하여 센서 네트워크 응용 소프트웨어의 효율적인 개발을 위해서는 핵심 모듈의 공통 기능과 수많은 센서 디바이스들의 다양한 역할 수행을 위한 가변기능을 구분하여 생산성을 높일 필요가 있다. 이를 위하여 센서 네트워크 응용 소프트웨어 개발에 프로덕트 라인 개발 기법을 적용하는 것이 효과적이다.

센서 네트워크 응용 소프트웨어 개발에 프로덕트 라인 개발 기법 적용을 통하여 도메인 공학에서 다양한 센서 네트워크 응용 도메인을 위한 공통피처를 식별, 핵심 자산을 구축하고, 응용 공학에서 도메인 공학의 결과들을 선택적으로 재사용하여 특정 센서들의 역할을 수행하는데 필요한 응용 소프트웨어를 효율적으로 개발할 수 있다.

2.2. 기존의 센서 네트워크 응용 소프트웨어 개발 기법

현재 센서 네트워크 응용 소프트웨어를 효율적으로 개발하기 위한 <표 1> 과 같은 다양한 기법 및 도구들이 연구

〈표 1〉 센서 네트워크 응용 소프트웨어 개발 기법

기법	내 용
TinyGALS[6]	이벤트 기반의 임베디드 시스템 프로그래밍을 위한 기법으로, 상위 수준의 명세를 이용하여 모델을 작성하고, 모델로부터 응용 소프트웨어를 자동으로 생성할 수 있는 기법
ATaG[7]	간단한 매크로를 이용하여 모델을 작성하고, 모델로부터 센서 네트워크 응용 소프트웨어를 자동으로 생성할 수 있도록 하는 기법
Regiment[8]	센서 네트워크를 위한 프로그래밍 언어로, 단지 몇 라인의 코드로 복잡한 센서 네트워크 응용 프로그램을 작성할 수 있도록 지원
SensorWare[9]	센서 노드에서의 계산, 의사소통, 감지 자원을 효율적으로 이용하도록 하는 경량의 모바일 제어 스크립트를 정의하고 지원하는 프레임워크
SNACK[10]	효율적인 센서 네트워크 응용 프로그램을 쉽게 개발할 수 있도록 하는 구성 언어, 컴포넌트, 서비스 라이브러리 및 컴파일러를 포함하는 개발 도구
Abstract regions[11]	하위 레벨의 통신, 데이터 공유, 오퍼레이션 등의 세부사항을 추상화하는 프로그래밍 원형을 제공함으로써 센서 네트워크 응용 프로그램의 설계를 간단하게 하는 기법
Kairos[12]	프로그래머가 자세한 분산 코드 생성이나 인스턴스화, 원격 데이터 접근과 관리, 노드 내의 프로그램 흐름 조정과 같은 것들을 자세히 알지 못하더라도 센서 네트워크 응용 프로그램을 구현할 수 있도록 하는 프로그래밍 기법
SPIDEY[13]	추상화된 언어를 제공함으로써 하위 레벨의 정보를 자세히 알지 못하더라도 센서 네트워크 응용 프로그램을 쉽게 작성할 수 있도록 지원

되어 활용되고 있다. TinyGALS[6]와 AtaG[7]는 모델로부터, 그 외의 기법들[8-13]은 추상화된 언어나 스크립트를 이용하여 작성한 프로그램으로부터 센서 네트워크 응용 소프트웨어를 자동으로 생성하기 위한 방법을 제공한다. 이러한 기존 방법들은 모델이나 추상화된 설계로부터 쉽고, 빠르게 센서 네트워크 응용 소프트웨어를 생성할 수 있게 해 준다는 장점이 있으나 단일 응용 소프트웨어 개발만을 고려한 방법이라는 한계를 지닌다.

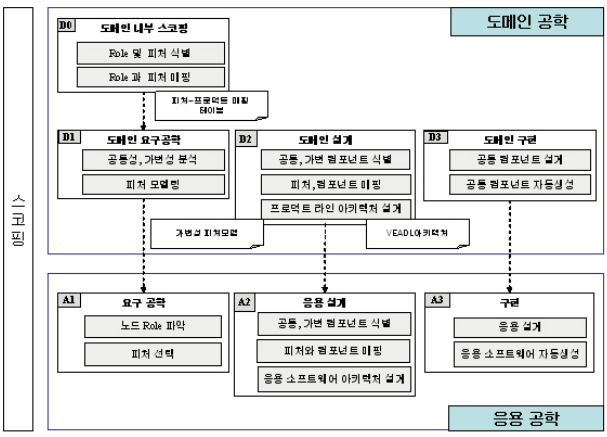
3. 프로덕트 라인 기반 센서 네트워크 응용 개발 프로세스

본 논문에서 제시하는 프로덕트 라인 기반 센서 네트워크 응용 소프트웨어를 개발 절차는 기존의 프로덕트 라인 개발 프로세스를 적용 도메인의 특성에 알맞게 커스터마이징하여 다음 (그림 1)과 같이 구성하였다.

센서 네트워크 응용 도메인의 경우 일반적인 비즈니스 도메인에 비하여 목표 플랫폼과 다양한 센서들로 인하여 도메인 스코핑에 중점을 두어야 하며, 가변성에 따라 포함될 수 있는 피처와 개발되는 소프트웨어의 종류가 달라질 수 있으므로 도메인의 가변성 표현에 중심을 두어야 한다. 따라서 본 논문에서 제시하는 개발 프로세스는 센서 네트워크 응용 도메인의 스코핑과 아키텍처 개발을 중점으로 하여 기존의 프로덕트 라인 개발 프로세스를 커스터마이징하였다. 기존의 프로덕트 라인 개발 기법에서는 도메인 공학에서 프로덕트 라인 스코핑을 따로 수행하지 않으나 (그림 1)의 제시 프로세스에서는 도메인 공학의 시작점으로 스코핑을 수행한다. 이는 이미 스코핑된 도메인 정의 내에서 다양한 센서의 역할에 따른 가변적성을 명확히 정의하기 위하여 수행된다. 또한 선택적으로 재사용되는 가변성을 포함하는 센서 네트워크 응용 소프트웨어의 프로덕트 라인 아키텍처 설계를 효율적으로 지원하기 위하여 가변성 피처 모델과 VEADL을 통한 아키텍처 설계를 수행하도록 구성하였다.

제시한 프로세스는 크게 도메인 공학과 응용 공학으로 구분된다. 도메인 공학에서는 센서 네트워크의 다양한 노드를 위한 응용 소프트웨어가 가져야 할 피처를 찾아내고, 공통된 부분을 구현하며, 응용 공학에서는 도메인 공학에서의 결과들을 재사용하여 특정 노드를 위한 응용 소프트웨어를 개발한다.

각 파트는 요구공학, 설계, 구현의 공통된 3단계로 구성되며, 도메인 공학에서는 요구공학 이전에 도메인 내부 스코핑 단계를 수행한다. 스코핑 단계에서는 피처와 응용 소프트웨어 종류와의 피처-프로덕트 매핑 테이블을 통하여 식별한 센서 네트워크 응용 소프트웨어의 피처들과 응용 소프트웨어 종류와의 관계를 모델링 할 수 있으며, 요구공학 단계에서는 센서 네트워크 응용 소프트웨어가 가지는 공통성과 가변성을 찾아내고, 가변성 피처 모델을 작성한다. 설계 단계에서는 제시한 방법을 통하여 센서 네트워크 응용 소프트웨어의 각 피처들을 구현 컴포넌트로 매핑하며, 이를 바탕으로 센서 네트워크 응용 소프트웨어에 대하여 VEADL을 통한 프로덕트 라인 아키텍처를 설계한다. 그리고, 구현 단계에서는 설계 단계에서 작성한 아키텍처에 따라 컴포넌트들의 상세 설계를 수행하고, 응용 소프트웨어가 수행될 운영체제와 구현 언어를 고려하여 센서 네트워크 응용 소프트웨어를 구현한다. 각 단계의 핵심내용을 정리하면 다음과 같다.



(그림 1) 센서 네트워크 응용 개발 프로세스

■ D0: 도메인 내부 스코핑

도메인 내부 스코핑 단계에서는 센서 네트워크의 다양한 노드의 응용 소프트웨어가 갖게 될 피처들을 파악하고, 응용 소프트웨어의 종류에 따라 피처들을 매핑한다. 하나의 센서 네트워크는 수많은 종류의 노드들로 구성되고, 센서 네트워크 도메인에 따라 노드들의 종류가 달라질 수 있으므로, 도메인 내부 스코핑을 통하여 프로덕트 라인에 포함될 응용 소프트웨어의 종류와 피처들을 파악하는 것이 필요하다. 이를 위하여 이 단계에서는 <표2>와 같은 응용 소프트웨어의 종류와 피처와의 관계를 보여주는 피처-프로덕트 매핑 테이블을 작성한다.

테이블의 각 행은 응용 소프트웨어의 피처를 나타내며, 각 열은 응용 소프트웨어의 종류를 나타낸다. 추후 이를 활용하여 공통성과 가변성을 추출한다.

■ D1: 도메인 요구공학

도메인 요구공학 단계에서는 도메인 내부 스코핑 단계(D0)에서 작성한 피처-프로덕트 매핑 테이블을 분석하여 응

〈표 2〉 피처-프로덕트 매핑 테이블

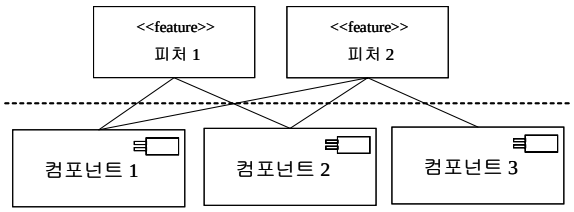
		소프트웨어 프로덕트1	소프트웨어 프로덕트2	소프트웨어 프로덕트3	소프트웨어 프로덕트4
피처 그룹1	피처 1		V	V	
	피처 2	V	V	V	
피처 그룹1	피처 3		V	V	
	피처 4		V	V	
	피처 5	V	V	V	V

용 소프트웨어 피처들의 공통성 및 가변성을 결정하고, 가변성 피처 모델을 작성한다.

본 논문에서는 기존의 특정 피처 모델을 차용하지 않고 UML기반의 확장된 가변성을 지원하는 가변성 피처 모델을 제시, 활용한다. 제시한 UML기반의 가변성 피처 모델은 다양한 사용자 층에 보편성을 제공하며, 또한 스테레오타입을 활용하여, 가변성 모델을 따로 작성해야 하는 일반적인 피처 모델과는 달리 가변점, 가변 피처 등의 다양한 가변성을 하나의 피처 모델 안에 표현할 수 있는 장점을 가진다. 다음 <표3>은 가변성 피처 모델에서 지원 가능한 다양한 가변성 표기법을 나타낸다. 가변성 피처 모델의 표기법은 Gomma[14]의 표기법을 바탕으로 사용자들이 쉽게 센서 네트워크 응용 소프트웨어에 대한 피처 모델을 작성할 수 있도록 스테레오타입을 확장 정의하였다.

■ D2: 도메인 설계

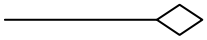
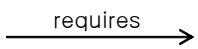
도메인 설계 단계에서는 공통의 피처에 대한 컴포넌트를 식별하고, 센서 네트워크 응용 소프트웨어에 대한 프로덕트 라인 아키텍처를 설계한다. 센서 네트워크 응용 소프트웨어를 구현하기 위해서는 응용 소프트웨어를 구성하는 컴포넌트를 식별하고, 응용 소프트웨어의 피처를 반영하여 각 컴포넌트를 설계해야 한다. 이를 위해서 센서 네트워크 응용 도메인 설계 단계에서는 센서 네트워크 응용 도메인 요구공학 단계에서 파악한 공통된 피처를 응용 소프트웨어의 컴포넌트로 매핑한다. (그림 2)는 피처를 컴포넌트로 매핑하는 방법을 보여준다. 하나의 피처는 하나 이상의 컴포넌트와 매핑이 되며, 하나의 컴포넌트는 하나 이상의 피처를 반영하여 설계하여야 한다.



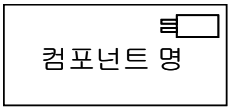
(그림 2) 피처와 컴포넌트의 매핑

공통된 피처를 컴포넌트로 매핑한 후에는 컴포넌트들을 구성하여 응용 소프트웨어의 프로덕트 라인 아키텍처를 설계한다. 이 단계에서는 가변성 피처 모델에서 제시된 가변성을 아키텍처 관점으로 일관성 있게 반영하기 위하여 기존

〈표 3〉 가변성 피처 모델의 표기법

표기법	설명
<<feature group type>> 피처그룹명	피처 그룹을 나타낸다. 특정 기능을 위한 피처들을 하나의 그룹으로 묶어서 표현할 때 사용한다.
<<feature type>> 피처명	각 피처 그룹에 속하는 하나의 피처를 나타낸다.
	피처 그룹과 피처 사이의 관계를 나타낸다.
«feature group type»	피처 그룹의 타입을 나타낸다. 피처 그룹의 타입으로는 «zero-or-one feature group», «exclusive-or feature group», «inclusive-or feature group» 이 있다.
«zero-or-one feature group»	필요에 따라 그룹에 속해 있는 피처가 선택되지 않거나 하나만 선택되어야 하는 경우를 나타낸다.
«exclusive-or feature group»	그룹에 속해 있는 피처 중에서 오직 하나만 선택되어야 하는 경우를 나타낸다.
«inclusive-or feature group»	필요에 따라 그룹에 속해 있는 피처가 선택되지 않거나 하나이상 선택되는 경우를 나타낸다.
«feature type»	피처의 타입을 나타낸다. 피처의 타입으로는 «optional», «alternative», «default» 가 있다.
«optional»	필요에 따라 포함되는 피처를 나타낸다.
«alternative»	선택적인 피처를 나타낸다.
«default»	기본으로 포함되는 피처를 나타낸다.
	피처 사이의 관계를 나타내는 것으로, 하나의 피처가 선택되면 같이 선택되어야 하는 피처 간의 관계를 나타낼 때 사용한다.
«variation point»	가변점을 나타낸다. 센서 네트워크 응용 소프트웨어의 특성에 따라 가변적인 피처들의 그룹을 표현할 때 사용한다.
«variant»	가변적인 피처를 나타낸다.

〈표 4〉 VEADL의 표기법

표기법	설명
	컴포넌트를 나타낸다. 센서 네트워크 응용 소프트웨어를 구현하는데 필요한 컴포넌트를 의미한다.
	센서 네트워크 응용 소프트웨어의 가변점을 나타낸다. 응용 소프트웨어를 구성하는 일부분이지만, 응용 소프트웨어의 종류에 따라 달라지는 부분으로 응용 공학 부분에서 특정 응용 소프트웨어를 위한 컴포넌트로 대체되어야 하는 부분이다.
	컴포넌트나 가변점 사이의 관계를 나타낸다. 서로 관련되어 있는 컴포넌트나 가변점을 연결하는데 사용한다.

의 ADL을 확장하여 가변성 표기를 추가한 VEADL을 제시하였다. <표 4>는 응용 소프트웨어의 프로덕트 라인 아키텍처를 설계하기 위해 필요한 VEADL의 표기법을 보여준다.

도메인 공학 단계에서는 센서 네트워크 응용 소프트웨어의 공통 부분에 대해서만 컴포넌트로 설계가 되고, 가변적인 부분에 대해서는 가변점을 사용하여 남겨둔다. 가변점에 해당하는 부분은 응용 공학 단계에서 실제 개발할 소프트웨어의 특성에 맞도록 선택한 피처를 반영하는 컴포넌트로 대체가 된다.

■ D3: 도메인 구현

도메인 구현 단계에서는 설계된 센서 네트워크 응용 소프트웨어의 프로덕트 라인 아키텍처의 공통 컴포넌트에 대한 상세설계를 하고, 구현한다. 응용 소프트웨어가 수행될 센서 네트워크 운영체제에 맞도록 컴포넌트를 설계하고, 운영체제와 호환되는 프로그래밍 언어를 사용하여 구현한다. 센서 네트워크 응용 소프트웨어를 설계 모델로부터 자동으로 생성하는 방법 및 도구가 다양하게 지원되므로, 이 단계에서는 공통된 컴포넌트에 대한 설계 모델로부터 컴포넌트의 소스 코드를 자동으로 생성한다.

■ A1: 요구공학

요구 공학 단계에서는 특정 역할을 수행하는데 필요한 센서 네트워크 응용 소프트웨어의 피처를 도메인 공학에서 작성한 피처와 응용 소프트웨어 종류와의 매핑 테이블로부터 선택한다. 선택한 피처와 도메인 공학에서 작성한 피처 모델을 바탕으로 개발하고자 하는 센서 네트워크 응용 소프트웨어를 위한 피처 모델을 작성한다.

■ A2: 응용 설계

응용 설계 단계에서는 구현하고자 하는 특정 센세 너트워크 응용 소프트웨어를 위한 가변적인 피처를 컴포넌트로 매핑하고, 아키텍처를 설계한다. 공통된 피처에 대해서는 도메인 공학에서 이미 컴포넌트로 매핑을 했으므로 재사용하고, 가변적인 피처를 구현하는데 필요한 컴포넌트를 식별하고, 피처를 컴포넌트로 매핑한다. 피처를 컴포넌트로 매핑한 후에는 도메인 공학에서 설계한 프로덕트 라인 아키텍처의 가변점 부분을 실제 구현할 컴포넌트로 대체한다. 이 때, 필요하지 않은 가변점은 삭제하여 응용 소프트웨어를 위한 아키텍처를 완성한다.

■ A3: 응용 구현

응용 구현 단계에서는 센서 네트워크 응용 소프트웨어 아키텍처의 가변적인 부분에 대한 컴포넌트들을 상세 설계하여 구현하고, 도메인 공학에서 개발한 공통 컴포넌트들과 연결하여 특정 역할을 수행하는데 필요한 센서 네트워크 응용 소프트웨어를 완성한다. 도메인 공학에서와 마찬가지로 컴포넌트의 설계 모델로부터 소스 코드를 자동으로 생성하여, 이미 구현된 공통 컴포넌트와 통합한다.

4. 적용 사례

이 장에서는 본 논문에서 제시하는 기법을 적용한 센서 네트워크 응용 소프트웨어 개발 사례를 제시한다. 제시한 센서 네트워크 응용 소프트웨어 개발 사례는 ETRI에서 개발한 Nano-Qplus[15] 운영체제를 기반으로 한다.

4.1. 도메인 공학

도메인 공학에서는 Nano-Qplus 기반의 센서 네트워크에서 필요로 하는 센서, 라우터, 싱크, 엑츄에이터 역할을 수행하는 노드들을 위한 응용 소프트웨어가 가져야 할 피처를 찾아내어 공통성 및 가변성을 분석하고, 공통된 부분을 구현한다.

4.1.1. D0: 도메인 내부 스코핑

센서 네트워크 응용 소프트웨어의 종류는 센서 네트워크에 존재하는 노드의 역할에 따라 구분할 수 있다. 센서 네트워크에는 크게 센서, 라우터, 싱크, 엑츄에이터 역할을 하는 노드들이 존재하므로 센서 네트워크 응용 소프트웨어의 종류 또한 크게 네 가지로 나눌 수 있다. 데이터 감지 소프트웨어는 주변 환경을 감지하여 이에 대한 데이터를 생성 전달하는 역할을 수행할 수 있도록 하며, 데이터 전달 소프트웨어는 데이터를 중계하는 역할을 수행하도록 한다. 데이

〈표 5〉 센서 네트워크 응용 소프트웨어의 피처-프로덕트 매핑 테이블

피처 \ 소프트웨어 프로덕트 종류		데이터 감지	데이터 전달	데이터 수집 /처리	구동
스케쥴러	FIFO	∨	∨	∨	∨
	Preemption-RR	∨	∨	∨	∨
UART	Printf	∨	∨	∨	∨
	Scanf	∨	∨	∨	∨
통신 프로토콜	Simple Send/Recv	∨	∨	∨	∨
	Star-Mesh route	∨	∨	∨	∨
네트워크	RF 채널		∨	∨	
	Scan 채널	∨			∨
	PAN coordinator		∨	∨	
RF 기능	연결		∨	∨	
	전송	∨	∨	∨	∨
	수신		∨	∨	∨
ADC	배터리 센서	∨			
	온도 센서	∨			
	조도 센서	∨			
	습도 센서	∨			
	가스 센서	∨			
	적외선 센서	∨			
	초음파 센서	∨			

터 수집/처리 소프트웨어는 감지된 모든 데이터들을 수집하여 센서 네트워크를 제어하는 역할을 수행하도록 하며, 구동 소프트웨어는 명령에 따라 특정 디바이스를 구동시키는 역할을 수행하도록 한다.

이와 같이 노드의 역할에 따라 응용 소프트웨어의 종류가 달라지므로, 각 노드의 역할에 따라 응용 소프트웨어에 포함되어야 하는 피처가 달라진다. 따라서 도메인 내부 스코핑 단계에서는 이러한 것들을 고려하여 센서 네트워크 응용 소프트웨어가 포함해야 하는 모든 필요한 피처를 추출한다. <표 5>은 Nano-Qplus 기반의 센서 네트워크 응용 소프트웨어가 가져야 하는 피처와 프로덕트 종류와의 관계를 보여준다. 피처들은 Nano-Qplus가 제공하는 기능들을 바탕으로 도출한 것이며, 소프트웨어 프로덕트의 종류는 Nano-Qplus에서 지원하는 센서, 라우터, 싱크, 엑츄에이터 기능을 수행하는 노드를 위한 응용 소프트웨어의 종류를 식별한 것이다.

4.1.2. D1: 도메인 요구공학

도메인 요구공학 단계는 도메인 내부 스코핑 단계에서 작성한 피처-프로덕트 매핑 테이블을 분석하여, 공통성과 가변성을 식별하고, 센서 네트워크 응용 소프트웨어를 개발하기 위한 가변피처 모델을 작성한다. <표 5>를 분석하여 다음과 같이 공통성과 가변성을 분석할 수 있다.

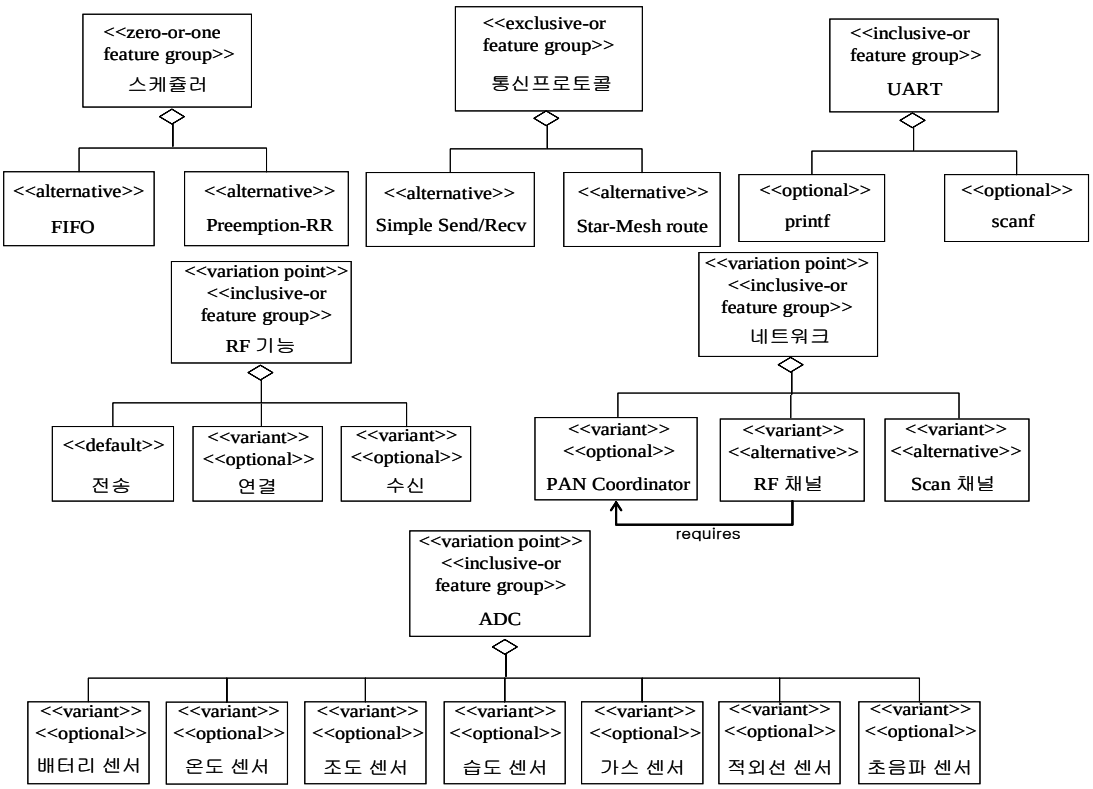
스케쥴러와 통신 프로토콜은 센서 네트워크를 수행하기 위해서 공통으로 필요한 피처로, 센서 네트워크 응용 소프트웨어를 개발할 때 항상 포함이 된다. 이 때, 상황에 따라 알맞은 스케쥴러와 통신 프로토콜을 선택하여 개발한다.

UART는 데이터 입출력을 위해 필요한 피처로, 응용 소프트웨어에서 처리한 데이터를 사용자가 볼 수 있도록 화면에 출력하거나 입력 받을 경우에 포함한다. 이 피처는 모든 종류의 소프트웨어를 개발할 때 필요에 따라 포함하거나 포함되지 않을 수 있는 공통된 피처이다.

ADC는 데이터를 감지하기 위해 필요한 피처로 데이터 감지 응용 소프트웨어를 개발할 경우에 포함한다.

네트워크는 소프트웨어의 종류에 따라 포함되는 피처가 다르다. RF 채널과 PAN cordinator 피처는 센서 네트워크 상에서의 데이터 전달과 수집에 관련된 피처로 데이터를 전달하거나 수집하는 것을 주 목적으로 하는 응용 소프트웨어를 개발할 경우에 포함되며, Scan 채널 피처는 데이터를 감지하고, 구동하는 역할을 주로 수행하는 응용 소프트웨어를 개발할 경우에 포함된다.

RF 기능 역시 소프트웨어의 종류에 따라 포함되는 피처가 다르다. 모든 센서 네트워크 응용 소프트웨어는 RF를 통하여 데이터를 전송할 수 있어야 하므로 전송 피처는 공통으로 필요한 피처이고, 연결 피처는 RF 통신을 위하여 다른



(그림 3) Nano-Qplus 기반의 센서 네트워크 응용 소프트웨어 개발을 위한 가변성 피처 모델

노드와의 연결을 담당하는 역할을 포함하는 응용 소프트웨어를 개발할 경우에 필요한 피처이며, 수신 피처는 RF를 통하여 데이터를 전달 받는 역할을 포함하는 응용 소프트웨어를 개발할 경우에 필요한 피처이다.

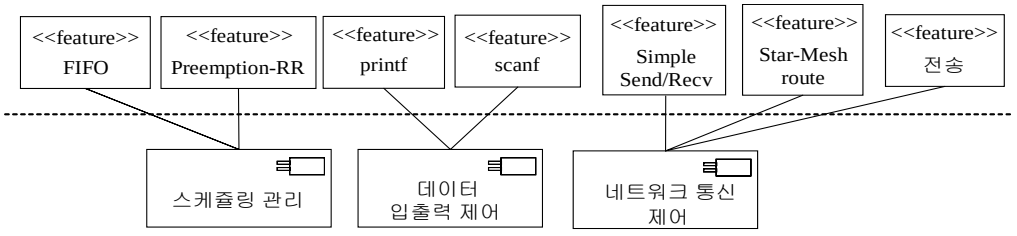
이와 같은 공통성 및 가변성 분석을 바탕으로 가변성 피처 모델을 작성한다. (그림 3)은 도메인 내부 스코핑 단계에서 도출한 <표 5>의 센서 네트워크 응용 소프트웨어의 피처들을 바탕으로 <표 3>의 가변성 피처 모델의 표기법을 이용하여 작성한 가변성 피처 모델이다. 노드의 역할에 따라 가변적인 피처들은 <<variant>> 스테레오 타입을 이용하여 표현하고, 이러한 피처를 포함하고 있는 피처그룹은 <<variation point>> 스테레오 타입을 이용하여 표현함으로써, Nano-Qplus 기반의 센서 네트워크 응용 소프트웨어를 위한 피처들의 공통성과 가변성을 명확히 알 수 있는 피처 모델을 작성하였다.

응용 공학에서는 이 피처 모델을 바탕으로 개발하고자 하는 소프트웨어의 종류에 따라 필요한 피처 그룹을 선택하고, 선택한 피처 그룹에 속해 있는 피처들 중에서 소프트웨어에 포함시킬 피처를 선택하여 구현하면 된다.

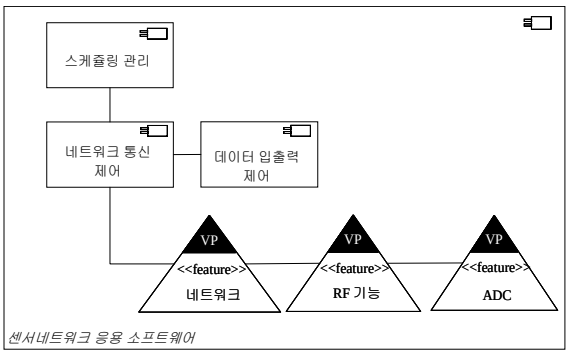
4.1.3. D2: 도메인 설계

센서 네트워크 응용 소프트웨어에 포함될 피처가 결정이 되면, 각 피처를 구현할 수 있는 컴포넌트를 식별하고, 식별한 컴포넌트를 선택한 피처와 매핑하는 것이 필요하다. (그림 4)는 센서 네트워크 응용 소프트웨어에 항상 필요한 공통된 피처를 구현에 필요한 컴포넌트로 매핑한 결과를 보여준다. 이전 단계에서 작성한 피처 모델을 바탕으로 7가지의 공통된 피처를 결정하고, 이를 구현하기 위한 스케쥴링 관리, 데이터 입출력 제어, 네트워크 통신 제어 컴포넌트를 식별하여 공통 피처와 매핑시켰다. 센서 네트워크 응용 소프트웨어 프로덕트 라인 아키텍처를 설계할 때, 공통된 피처에 대해서는 (그림 4)의 컴포넌트를 사용하면 된다.

공통된 피처와 컴포넌트를 매핑한 후, 컴포넌트들을 구성하여 VEADL을 이용한 응용 소프트웨어의 프로덕트 라인



(그림 4) 공통 피처와 컴포넌트의 매핑



(그림 5) VEADL을 이용한 프로덕트 라인 아키텍처 명세

아키텍처를 설계한다.

(그림 5)는 센서 네트워크 응용 소프트웨어의 프로덕트 라인 아키텍처를 보여준다. 공통된 피처에 대해서는 (그림 4)의 컴포넌트를 사용하고, 가변적인 피처에 대해서는 가변점을 이용하여 명세하였다. 컴포넌트 간의 연결선은 상호 연관관계를 나타낸다.

이러한 프로덕트 라인 아키텍처를 이용하면, 소프트웨어의 종류에 따라 해당 컴포넌트만을 선택하여 응용 소프트웨어의 아키텍처를 설계할 수 있다. 개발하고자 하는 소프트웨어의 종류에 따라 가변점에 있는 가변적인 피처 중에서 필요한 것만을 선택하여 응용 소프트웨어를 설계하면 된다.

4.1.4. D3: 도메인 구현

센서 네트워크 응용 소프트웨어의 프로덕트 라인 아키텍처가 설계되면, 센서 네트워크 응용 도메인 구현 단계에서는 공통된 부분에 대해서 상세설계를 하고, 구현한다. 본 논문의 예제에서는 (그림 5)에 나타나 있는 스케쥴링 관리, 네트워크 통신 제어, 데이터 입출력 제어의 3개의 컴포넌트에 대해서 상세설계를 하고, 구현을 한다. 센서 네트워크 응용 소프트웨어의 컴포넌트에 대한 상세설계를 수행할 때에는, 응용 소프트웨어가 탑재될 센서 네트워크 운영체제에 적합하도록 설계한다. 본 논문에서는 Nano-Qplus를 기반으로 응용 소프트웨어가 구현될 수 있도록 설계한다. 컴포넌트를 설계한 후에는 설계 모델로부터 Nano-Qplus 운영체제에서 수행될 수 있는 다양한 방법을 이용하여 소스 코드를 자동으로 생성한다.

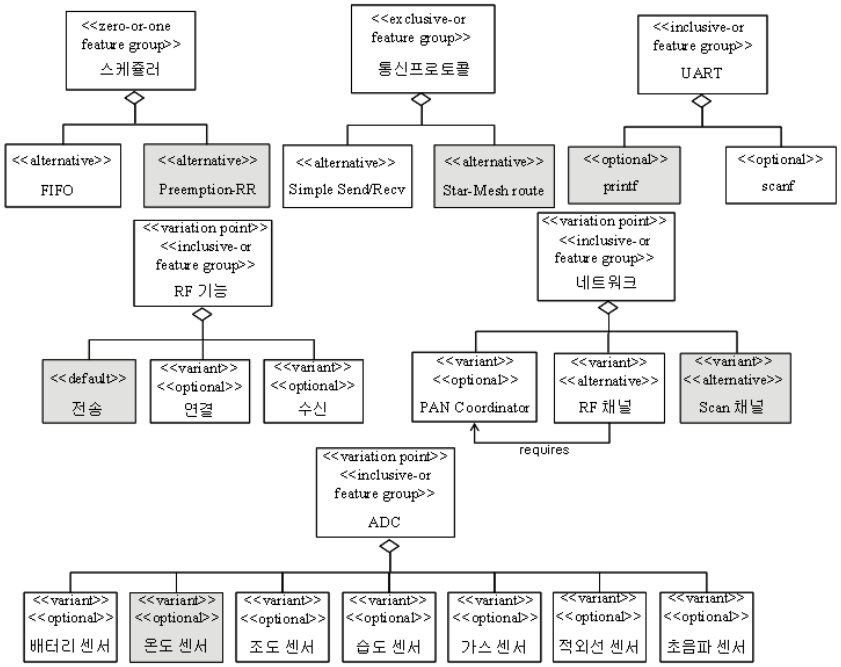
4.2. 응용 공학

이 절에서는 도메인 공학 단계에서 개발한 센서 네트워크 응용 소프트웨어 프로덕트 라인의 산출물들을 활용하여 홈 네트워크 시스템에서 실내 온도를 감지하는 역할을 수행하는 센서 노드에 대한 응용 소프트웨어의 개발 프로세스 중 아키텍처 설계까지를 보여준다.

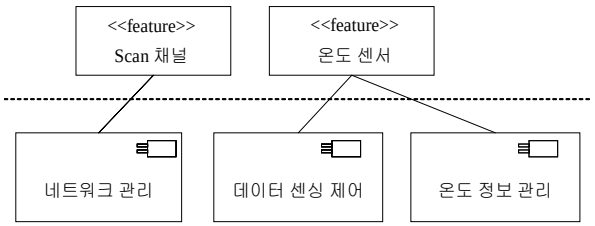
(그림 6)은 온도 센서 노드를 위한 응용 소프트웨어의 피처를 선택하기 위한 피처 모델링을 보여준다. 회색으로 되어 있는 부분이 온도 센서 역할을 수행하는 노드를 위한 응용 소프트웨어를 개발하기 위해 선택된 피처를 나타낸다.

(그림 7)은 (그림 6)의 피처 모델에서 선택한 피처 중 가변적인 피처를 컴포넌트로 매핑하여 명세하였다. 공통된 피처에 대해서는 도메인 공학에서 컴포넌트로 매핑을 수행하고, 응용 공학에서는 각 응용 소프트웨어를 위한 가변적인 피처에 대해 컴포넌트를 도출하고 매핑을 수행한다. (그림 6)에서 선택한 피처에 따라 프로덕트 라인 아키텍처에서 응용 소프트웨어 아키텍처에 포함될 가변점을 선택하고, 피처와 컴포넌트 간의 매핑을 통하여 온도 센서 노드를 위한 응용 소프트웨어의 아키텍처를 설계할 수 있다.

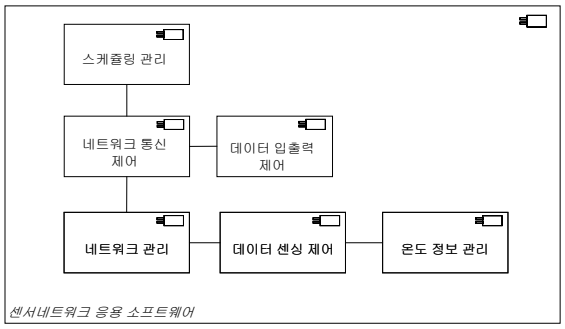
(그림 8)은 제시하는 기법에 따라 온도 센서 노드를 위해



(그림 6) 온도 센서를 위한 응용 소프트웨어의 가변성 피처 모델



(그림 7) 온도 센서의 응용 소프트웨어를 위한 피처와 컴포넌트의 매핑



(그림 8) 온도 센서의 응용 소프트웨어를 위한 아키텍처 명세

설계된 응용 소프트웨어의 아키텍처를 나타낸다. 이 아키텍처는 (그림 5)의 프로덕트 라인 아키텍처를 바탕으로 가변점 부분을 (그림 7) 컴포넌트로 대체하여 작성한 것이다. 프로덕트 라인 아키텍처의 네트워크, ADC 가변점은 (그림 7)의 컴포넌트도 대체하고, RF 기능의 가변점에 대해서는 선택된 가변 피처가 없으므로 프로덕트 라인 아키텍처로부터 삭제하였다.

이와 같이 설계된 온도 센서 노드를 위한 응용 소프트웨어는 공통 컴포넌트인 스케쥴링 관리, 네트워크 통신 제어, 데이터 입출력 제어 컴포넌트와 함께 온도 감지를 제어하고, 감지한 데이터를 관리하고, 전송하기 위해 필요한 데이터 센싱 제어, 온도 정보 관리 및 네트워크 관리 컴포넌트를 포함한다.

5. 결 론

현재 센서 네트워크 응용 소프트웨어를 개발하기 위한 다양한 방법들은 단일 센서 네트워크 응용 소프트웨어를 빠르게 개발하기 위해 모델이나 추상화된 설계로부터 자동적인 코드를 생성하는데 관심을 두고 있다. 그러나 이러한 기존의 방법으로는 핵심공통 모듈의 재사용을 통한 개발 생산성을 극대화하기는 어렵다.

이러한 한계점을 극복하기 위하여 본 논문에서는 프로덕트 라인 기법을 센서 네트워크 응용 소프트웨어 개발에 적용, 도메인 공학에서 공통성과 가변성 분석을 통하여 식별된 피처와 센서 네트워크 아키텍처를 기반으로 핵심 자산을 구축하고 응용 공학을 통하여 선택적으로 핵심 자산을 재사용하여 특정 센서 네트워크 응용 소프트웨어를 구축하는 사례를 제시하였다.

이를 위하여 프로덕트 라인 기법에 센서 네트워크 응용 도메인의 특성을 반영하여 프로세스를 커스터마이징하였으며, 도메인 공학에서 센서 네트워크 응용 소프트웨어의 스코핑을 효과적으로 수행하기 위한 피처-프로덕트 매핑 테이블을 제시하고, 센서 네트워크 응용 소프트웨어의 가변적 피처의 다양성을 표기하기 위한 UML기반의 가변성 피처 모델을 제시하였다. 또한 아키텍처 관점에서 선택적으로 재사용되는 가변성을 표현하기 위해 기존의 ADL을 확장하는 VEADL을 제시하였다. 본 논문이 센서 네트워크 응용 소프트웨어의 개발에 기여하는 바는 다음과 같다.

첫째, 일반적인 프로덕트 라인 개발 프로세스를 스코핑과 아키텍처의 구성이 중요시되는 센서 네트워크 응용 도메인의 특성에 맞게 커스터마이징하였다. 일반적인 프로덕트 라인 개발 프로세스와는 달리 도메인 공학의 시작점으로 도메인 내부 스코핑을 수행하도록 조정하였으며, 선택적으로 재사용되는 가변성을 포함하는 센서 네트워크 응용 소프트웨어의 프로덕트 라인 아키텍처를 설계하는 것에 초점을 두어 요구공학과 설계를 수행하도록 프로세스를 조정하였다. 이는 센서 네트워크 응용 도메인의 특성 즉, 다양한 센서와 센서 노드들의 한정된 자원에 의하여 응용 프로그램의 기능에 따라 핵심 모듈의 선택적 적재의 특성을 반영하기 위함이다.

둘째, 프로덕트 라인 개발 프로세스에 따라 효과적으로 센서 네트워크 응용 소프트웨어를 개발할 수 있도록 여러 산출물을 생성하기 위한 모델 및 표기법을 제시하였다. 제시된 피처-프로덕트 매핑 테이블은 응용 도메인의 스코프에 대한 가시성을 제공하며, 제시된 가변성 피처 모델의 표기법은 다양한 프로덕트 라인 기법에서 제시하는 특정 모델을 사용하지 않고 국제표준인 UML을 활용하여 보편성을 제공하며, 스테레오타입을 활용하여 가변점과 가변 피처를 고려한 가변성 표현의 확장을 제공한다. 또한 아키텍처 관점에서도 이를 일관성 있게 반영하기 위하여 기존의 일반적인 ADL을 확장하는 VEADL(Variability Extended ADL)을 제시하였다.

셋째, 사례 연구를 통하여 제시하는 프로세스 및 표기법이 센서 네트워크 응용 도메인의 핵심 자산 구축과 핵심 자산의 선택적 재사용에 효율적으로 적용가능하며 이를 통하여 센서 네트워크를 구성하는 다양한 노드를 위한 최적의 응용 소프트웨어를 개발할 수 있음을 보였다.

향후에는 제시한 프로세스의 세부 절차를 구성하고, 응용공학 관점에서 기존 방법들과의 연계 방안에 대하여 연구하고자 한다.

참 고 문 헌

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless Sensor Networks: A Survey,” Computer Networks, 38(4):393 - 422, 2002.

[2] Shigeru Fukunaga, Tadamichi Tagawa, Kiyoshi Fukui, Koichi Tanimoto, and Hideaki Kanno, “Development of ubiquitous sensor network,” Oki Technical Review, vol.

71, no. 4, pp. 24-29, 2004.
[3] Klaus Pohl, Gunter Bockle, Frank van der Linden, 'Software Product Line Engineering: Foundations, Principles, and Techniques,' Springer, 2005.
[4] Kyo C. Kang, Jaejoon Lee, and Donohoe, P., "Feature Oriented Product Line Engineering," IEEE Software, Vol. 9, No.4, pp.58-65, 2002.
[5] Bayer, j.,Flege, O., Knauber, P., Laqua, R., Muthig, D., Schmid, K., Widen, T., and Debaud, J., "PuLSE: A Methodology to develop Software Product Lines," Symposium for software Reusability 99, 1999.
[6] E. Cheong, J. Liebman, J. Liu, and F. Zhao, "Tinygals: a programming model for event-driven embedded systems," The Eighteenth Annual ACM Symposium on Applied Computing (SAC 2003), 2003.
[7] Bakshi, A., Prasanna, V.K., Reich, J., Larner, D., "The abstract task graph: A methodology for architecture-independent programming of networked sensor systems," 2005 Workshop End-to-end, Sense-and-Respond Systems, Applications and Services (EESR'05), pp. 19-24, 2005.
[8] R. Newton and M. Welsh, "Region streams: Functional macroprogramming for sensor networks," International Workshop on Data Management for Sensor Networks (DMSN 2004), 2004.
[9] A. Boulis, C. Han, and M. B. Srivastava, "Design and implementation of a framework for efficient and programmable sensor networks," First International Conference on Mobile Systems, Applications, and Services (MobiSys2003), 2003.
[10] B. Greenstein, E. Kohler, and D. Estrin, "A sensor network application construction kit (SNACK)," ACM SenSys'04, 2004.
[11] M. Welsh and G. Mainland, "Programming sensor networks using abstract regions," First Symposium on Networked Systems Design and Implementation (NSDI 2004), 2004.
[12] Ramakrishna Gummadi, Omprakash Gnawali, and Ramesh Govindan, "Macro-programming Wireless Sensor Networks Using Kairos," The 2005 International Conference on Distributed Computing in Sensor Systems (DCOSS '05), LNCS 3560, pp. 126 - 140, 2005.
[13] Luca Mottola and Gian Pietro Picco, "Logical Neighborhoods: A Programming Abstraction for Wireless Sensor Networks," The 2006 International Conference on Distributed Computing in Sensor Systems (DCOSS '06), LNCS 4026, pp.150 - 168, 2006.
[14] Hassan Gomaa, 'Designing Software Product Lines with UML,' Addison-Wesley Professional, 2004.
[15] Kwangyong Lee et al., "A Design of Sensor Network System based on Scalable & Reconfigurable Nano-OS Platform," IT-SoC2004, 2004.

김 영 희

e-mail : kyhse@ssuci.ac.kr
1974년 숭실대학교 전자계산학과 (학사)
1986년 숭실대학교 산업대학원 전자계산학과(석사)
2005년 숭실대학교 대학원 컴퓨터학과 (공학박사)
1973년~1978년 한국은행 사무개선부 전자계산과
1979년~현재 숭실대학교 전산원 소프트웨어정보학과 교수
관심분야 : 개발 프로세스, 방법론, 모델링, 유비쿼터스 컴퓨팅 등

이 우 진

e-mail : wjlee@icu.ac.kr
2000년 숭실대학교 컴퓨터학부(학사)
2002년 숭실대학교 대학원 컴퓨터학과 (공학석사)
2007년 숭실대학교 대학원 컴퓨터학과 (공학박사)
2007년~현재 한국정보통신대학교 공학부(Postdoc)
관심분야 : 유비쿼터스 컴퓨팅, 임베디드 시스템, SOA, 프로덕트 라인

최 일 우

e-mail : iwchoi@kangnam.ac.kr
1995년 숭실대학교 전자계산학과 (학사)
1997년 숭실대학교 대학원 컴퓨터학과 (공학석사)
2004년 숭실대학교 대학원 컴퓨터학과 (공학박사)
2004년~2006년 바산네트워크(주) 연구소장
2007년~현재 강남대학교 교양교수부 교수
관심분야 : 개발 프로세스, 레거시 재사용, 프로덕트 라인, SOA 등