시스템 엔지니어링 학술지 제1권 2호. 2005.7

MBASE를 적용한 유비쿼터스 컴퓨팅 시스템 개발 방법론

김상수, 이동현, 인 호*
고려대학교 컴퓨터학과

A MBASE-based Development Method for Ubiquitous Computing Systems

Sangsoo Kim, Donghyun Lee and Hoh In*

Department of Computer Science & Engineering, Korea University, 5-ga, Anam-dong, Sungbuk-gu, Seoul 136-701, Korea

Abstract : Ubiquitous computing is an emerging technic for various areas such as public, private (individual), military, industrial, medical sectors. The applications of ubiquitous computing are expected to be prevailed from home to office. Unfortunately, it is not effective to apply existing system/software development methodologies into this emerging ubiquitous computing applications. In this paper, MBASE (Model-Based system Architecting and Software Engineering) is applied as a development method of ubiquitous computing applications. The advantage of MBASE is to identify mismatches of product, process, property, and success models and resolve them in developing the ubiquitous computing applications. A SmartView, a ubiquitous computing application, is presented as a case study of MBASE.

Key Words : MBASE, Ubiquitous Computing, System Engineering Process, Requirements Engineering, Model Clash, SmartView.

1. 서 론

모든 정보가 언제어디서든지 자유롭게 흘러 다니는 유비쿼터스 사회가 도래되었다. 유비쿼터스 컴퓨팅은 사용자가 컴퓨터나 네트워크를 의식하지 않은 상태에서 장소에 구애받지 않고 자유롭게 네트워크를 접속할 수 있는 환경을 의미한다. 유비쿼터스 컴퓨팅은 현재 국방 무기체계 및 자동화 정보체계를 포함한 공공분야 뿐만 아니라 자동차, 항공기, 의료기기, 생활가전, 모바일 등 다양한 분야의 많은 제품들을 통해서 실현되고 있다. 향후에는 더욱더 유비쿼터스 컴퓨팅의 비중은 증가할 것이다. 이러한 유비쿼터스

* 교신저자 : hoh_in@korea.ac.kr

컴퓨팅 제품들의 주요 성공요소는 고품질의 서비스를 제공할 수 있는 시스템을 필요한 시기와 제한된 비용으로 성공적으로 개발하는 것이다. 이를 위해있는 시스템을 필요한 시기와 제한된 비용서는 시간과, 생산성, 비용, 품질의 최적화된 유비쿼터스 컴퓨팅 시스템을 개발하는 것이다.

유비쿼터스 컴퓨팅 시스템은 다양한 플랫폼이 복합된 시스템으로 구성되어 있다. 이러한 특성상 유비쿼터스 컴퓨팅 시스템을 개발하기 위해서는 일반적인 소프트웨어나 하드웨어 개발방법론을 통해서 개발하기는 부족함이 있다. 또한 유비쿼터스 컴퓨팅 시스템의 특수한 요구를 만족할 수 있는 방법론이 보편화 되어있지 않을 뿐만 아니라 시스템엔지니어링 프로세스를 적용하더라도 이러한 문제점을 극복하기는 힘들 것이다. 특히 유비쿼터스 컴퓨팅 도메인의 경우 구현의 성공요소는 다양한 플랫폼으로 구성되어 서로 이질적인 시스템간의 상호운용성(interoperability) 첨단시스템에 사용될 시 신뢰성(reliability factor), 기존 시스템을 활용한 복합시스템 구성, 비용(cost factor) 및 시장 출시 시간에 좌우되는 특징이 있으며 이러한 특수한 도메인을 겨냥한 개발 기술이 필요하다.

본 연구에서는 유비쿼터스 컴퓨팅 시스템 도메인에서 적용될 수 있는 개발방법론을 제안하였다. 본 논문에서 제안된 방법론은 유비쿼터스 컴퓨팅 시스템의 개발 수명주기 동안 발생할 수 있는 충돌로 인한 프로젝트의 실패를 방지하고 유비쿼터스 컴퓨팅 시스템 분야에 맞게 특성화하기에 용이한 MBASE(Model- Based system Architecting and Software Engineering)[1]를 기반으로 하고 있다.

MBASE 방법론을 유비쿼터스 컴퓨팅 시스템 개발에 적용하여 유비쿼터스 컴퓨팅 도메인의 특징인 상호운용성, 신뢰성, 재사용성, 비용 등과 관련된 모델간의 충돌을 찾아내고 이를 방지할 수 있는 개발방법론들을 중심으로 제안하였다. 이를 위해 본 논문의 2장에서는 MBASE방법론을 소개하고, 3장에서는 MBASE를 적용한 유비쿼터스 컴퓨팅 시스템 개발 방법론을 제시하고 결과를 도출한다. 4장에서는 유비쿼터스 컴퓨팅 시스템 개발사례를 제시하고, 5장에 결론과 향후 연구내용을 제시한다.

2. 배경: MBASE

MBASE는 프로세스(Process), 제품(Product), 특성(Property), 성공(P개cess) 모델을 통합적으로 관리하는 접근법으로 1998년 USC (University of South California)의 DR. Barry Boehm의 주도하에 저자 중 한 사람도 참여하여 개발하였다[2]. 시스템 개발 시 발생할 수 있는 모델들 간의 충돌을 방지하여 성공적으로 개발이 완료될 수 있도록 하는 접근법으로 시스템 개발에 필요한 전 수명주기간의 프로세스를 제공한다. USC CSE (Center for Software Engineering)에서는 매년 15개 이상의 프로젝트에 MBASE 방법을 적용하고 있으며, 미 공군, 국방성 등 다양한 프로젝트에 적용되고 있다.

소프트웨어 또는 시스템 아키텍처가 만족하는지 결정하기 위하여 컴포넌트, 커넥터, 형상 및 제약사항에 대한 명세서 이외에 반드시 필요한 것이 있다. 아키텍처 하나만으로는 이것의 완전성을 평가하기에는 상당한 불합리한 점이 있다. MBASE는 이러한 쟁점들을 해결하기 위해 제시된 시스템 개발 접근법이다. MBASE는 시스템 개발 프로젝트의 제품(Product), 프로세스(Process), 특성(Property), 성공(Puccess) 모델의 일치성과 상호작용에 초점을 맞추고 있다.

개발자들은 시스템을 개발하기 위해 수많은 개발 process를 사용하고 있다. 그중 가장 대표적인 것들로는 Waterfall, Evolutionary, Incremental, Spiral 모델 등이 있다. Product 모델은 운용개념,

요구사항, 아키텍처, 설계 및 코딩 등과 이들 간의 연관관계를 명세화 하는 것이다. Property 모델은 요구 및 적절한 수준과비용, 일정, 성능, 신뢰성, 재사용성, 보안 등의 품질속성과 같은 절충 가능한 프로젝트 요소로 정의된다. Success 모델은 Correctness, Stakeholder Win-Win, Business -case, IKIWISI(I'll know it when I see it), Mission completeness 등이 있다[3].

시스템 개발과정에서 발생할 수 있는 일반적인 모델충돌은 다음과 같다.

- Design-to-schedule 프로세스와 우선순위가 설정되지 않은 요구사항 간의 충돌
- COTS(Commercial Off The Shelf) - driven product와 Waterfall process 간의 충돌
- Risk-based process와 Spec-based progress payments 간의 충돌
- Life-cycle architecture 없이 Evolutionary 방식으로 개발 시
- Golden rule과 Stakeholder Win-Win 방식 간의 충돌
- Spec-based process와 IKIWISI Success model

MBASE 개발 방법을 개발프로젝트에 따라 발생 가능한 충돌을 방지하기 위하여 다양한 방법으로 모델을 조합하여 적용할 수 있다. MBASE 개발방법을 구성하는 근간은 Stakeholder Win-Win(Success model), Win-Win Spiral model(Process model), LCO 및 LCA 산출물 명세서(Product model), 주요 지점별 Milestones (Property model)이다. 특히, MBASE에서는 도메인의 특성에 맞는 접근방법과, 프로젝트의 특정 도메인에 적합하도록 첫 번째 단계의 모델 레이어로 부터 출발하는 일관적이고 통합된 도메인 묘사 방식을 적용하고, 시스템 수명주기 프로세스 간에 유연성을 제공, 참조 모델을 활용하고 모델기반으로 개발하는데 적합한 접근 방법 등 일발적인 다른 객체지향 접근법과 차별화 된 객체지향 개발방법을 적용하고 있다.

3. 유비쿼터스를 위한 MBASE 개발 방법론의 적용

3.1 유비쿼터스 컴퓨팅 도메인 분석

유비쿼터스 컴퓨팅 시스템은 일반적인 시스템과 다른 다양한 특징들 때문에 유비쿼터스 컴퓨팅 시스템의 개발은 일반적인 시스템개발과는 많은 차이점을 가지고 있다. 유비쿼터스 컴퓨팅 시스템 구현 기술은 네트워크의 고도화, 컨버전스(convergence), 광대역화 , 정보기기의 저가격화 등이 이루어지지 않으면 안 되는 특징을 가지고 있으며, 정보기기에 컴퓨팅과 통신기능의 추가, 인간화된 인터페이스(calm technology)로서 인간의 눈에 보이지 않아야하며, 상황에 따라 제공하는 서비스가 변해야 한다. 또한 임베디드 시스템의 일반적인 특징인 hard timing 제약, 메모리 및 전원사용의 한계, 규정된 하드웨어 플랫폼 기술, 시스템 비용과 같은 특수한 제약사항에 역점을 두어야 한다.

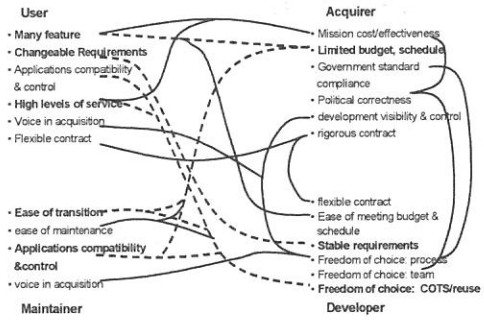

Fig. 1. Model Clash in Ubiquitous Systems

Fig. 1은 MBSAE에서 제시하는 방법에 따라 유비쿼터스 컴퓨팅 시스템 개발 과정에서 발생할 수 있는 모델 간의 충돌을 찾아낸 것이다. 유비쿼터스 컴퓨팅 시스템의 주요한 특징인 비용, 신뢰성을 포함한 높은 품질, 플랫폼 기술 등 제약사항, COTS 및 legacy 시스템의 재사용 등과 관련된 충돌을 찾아냈다.

Table 1. Model clash factors

Clash	Stakeholder	Phase
Property-Product	User, Developer	Requirements
Product-Property	User, Acquirer	Architecture
Product-Property	User, Developer, Maintainer	Implement
Product-Success	User, Developer, Maintainer	Implement
process-Property	User, Developer	Integration

Table 1은 유비쿼터스 컴퓨팅 시스템 개발시 발생할 수 있는 주요 충돌 내용을 충돌형태, 충돌과 관련된 이해당사자 그리고 주로 발생되는 시점별로 구분하여 나타낸 것이다. 유비쿼터스 컴퓨팅시스템 개발시 방생하는 충돌들은 일반적인 소프트웨어를 포함한 시스템 및 개발시 나타나는 충돌들을 거의 대부분 포함하고 있었으며 유비쿼터스 컴퓨팅 시스템의 특성 때문에 갖는 충돌들도 찾아낼 수 있었다.

3.2 모델충돌 해결방안

3.2.1 Product-Property-Success 모델충돌해결방안

획득부서는 제한된 개발비용과 일정을 고려하지만 사용자는 다양한 기능을 원하고 개발자는 안정된 요구사항을 근거로 개발을 하기를 원하기 때문에 이들 간에 충돌이 일어날 수 있다.

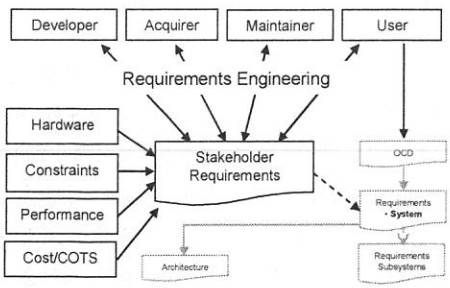

Fig. 2. Requirements generation process

대부분의 소프트웨어 개발에서처럼 유비쿼터스 컴퓨팅 시스템 개발에는 다양한 이해당사자(stakeholder)들이 존재한다. 이것은 시스템엔지니어링 프로세스를 수행하는 동안 확연하게 드러난다. 사용자는 요구사항의 잦은 변경을 원하고 개발자는 요구사항이 안정된 상태에서의 개발을 원한다. Fig. 2에 제시한 바와 같이 이해당사자와 주변 환경을 종합적으로 고려한 요구공학 절차를 적용하여 단순한 사용자 요구사항이 아닌 이해당사자 요구사항(stakeholder requirements)을 개발해야 한다.

MBASE에서의 Success 모델은 이해당사자들 모두가 win 할 수 있는 접근법을 제공하고 있다. 개발 프로젝트의 성공 요구사항은 어느 하나의 이해당사자만의 만족해서는 안된다. 예를 들어 사용자 측면에서는 성능과 일정 면에서 제품에 만족하더라도 획득기관과, 개발자가 비용적인 측면에서 만족하지 않는다면 프로젝트가 성공했다고 볼 수 없는 것이다[4]. 모든 이해당사자들이 win 할 수 있도록 하기 위해서는 개발의 전 과정에서 요구사항을 이해당사자 요구사항으로 개발하고 관리해야 한다. 이를 위하여 WinWin negotiation model을 적용하여 이해당사자들 간의 충돌(conflict)을 효과적으로 해결할 수 있다.

3.2.2 Product-Success 모델충돌 해결방안

유비쿼터스 컴퓨팅 시스템에서는 기존의 정보기기를 사용하여 새로운 시스템과 통합하는 경우가 많다. 이때 개발자가 COTS를 선택하여 개발할 경우 사용자 및 시스템관리자가 원하는 적용성과 적합성과의 충돌이 발생한다.

유비쿼터스 컴퓨팅 시스템에서의 소프트웨어 개발의 경우 하드웨어 플랫폼 기반으로 개발되어야 하기 때문에 특정 제품 또는 시스템에 맞게 소프트웨어를 개발해야 하는 제약사항을 갖는다. 이러한 경우 개발 프로세스의 적용에 따라 달라질 수 있다. COTS를 사용하는 경우

Waterfall 프로세스를 사용할 경우에는 충돌을 발생 시킬 수 있다. MBASE 개발 방론에서는 COTS 및 legacy 시스템을 활용하기 적합한 접근법으로 WinWin Spiral 모델 등을 제공하고 있다. 요구사항의 정의 정도와 변화에 따라 대응할 수 있는 방법론을 적용해야만 유비쿼터스 컴퓨팅 시스템의 특성에서 발생할 수 있는 문제점을 해결할 수 있다.

3.2.3 Process-Property 모델충돌 해결방안

유비쿼터스 컴퓨팅시스템은 각각의 플랫폼 간에 또는 소프트웨어 하드웨어간의 독립적인 개체로서 인증될 수 없다는 것은 중요한 사실이다. 유비쿼터스 컴퓨팅 시스템의 개발 프로세스를 결정하는 가장 중요한 요소로 작용하기 때문이다. 소프트웨어의 경우 하드웨어 시스템의 개발과 병행하여 개발되며 상호간에 정보를 교환하는 방식으로 개발이 진행되어야 한다. Fig. 3에서와 같이 시스템 레벨에서의 각 절차간의 절충 과정은 개발 수명주기 간 지속적으로 이루어져야 한다. Co-design 접근법은 일반적인 개발방법에서 설계단계부터 하드웨어와 소프트웨어를 분리하여 설계하는 방식이 아니라 상호 제약사항과 요구사항을 보완해 가면서 동일한 프로세스에서 개발하는 방법으로 유비쿼터스 컴퓨팅 시스템 개발시에 적절한 접근법이다.

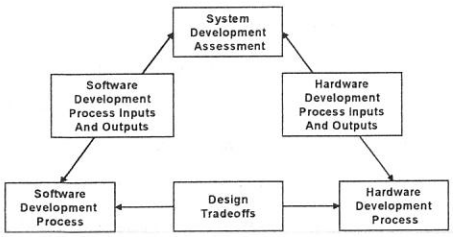

Fig. 3. Relation between hardware and software development process

4. 유비쿼터스 컴퓨팅 시스템 개발사례

4.1 스마트 뷰 운용개념

본 장에서는 유비쿼터스 컴퓨팅 시스템에서 발생할 수 있는 다양한 충돌을 해결하기 위하여 MBASE를 적용한 사례를 제시 하고자 한다. 스마트 뷰(SmartView)는 유비쿼터스 컴퓨팅 환경에 필요한 정황인지 미들웨어 아키텍처를 기반으로 휴대용 모바일 플랫폼과 PC, DVD 플레이어 등을 이용하여 정황인지 크로스 플랫폼 어플리케이션[5]을 구현한 것이다. 스마트 뷰는 유비쿼터스 환경에서 언제 어디서나 누구나 엔터테인먼트를 즐길 수 있는 시스템의 구축의 대표적인 예로, 고려대학교 컴퓨터학과 임베디드 소프트웨어공학 연구실에서 개발하였다.

유비쿼터스 엔터테인먼트 서비스 시스템은 사용자에게 언제 어디서나 엔터테인먼트 서비스를 즐길 수 있도록 해주는 것이다[6]. 시스템은 일반적으로 모바일 플랫폼과 가정 및 사무실의 고정용 플랫폼 간에 정황인지 미들웨어를 기반으로 한 크로스 서비스 어플리케이션이다. Fig. 4는 유비쿼터스 엔터테인먼트 서비스 시스템 운용개념도이다.

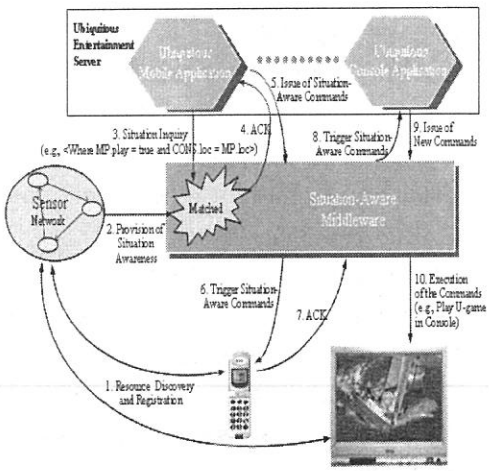

Fig. 4. Ubiquitous entertainment service overview

4.2 스마트 뷰 구현

스마트 뷰는 유비쿼터스 엔터테인먼트 서비스 시스템의 프로토타입으로 개발된 것으로, PC와 PDA사이에서 크로스 플랫폼 영화와 게임으로 구현되었다. 센서 모트가 PDA에 부착되어 있다. 사용자가 PDA에서 영화를 감상하고 있다가 PC에 접근하면, PC는 연결된 모트의 게이트웨이를 통해서 사용자가 접근했다는 정보를 받게 되고, PDA에서 재생되고 있던 영화를 끊김 없이 PC로 옮긴다. 그리고 사용자가 TV로 영화를 보다가 PDA를 가지고 밖으로 나가게 되면, 영화는 다시 PDA에서 재생된다. Fig. 5는 구현된 결과를 보여주고 있다.

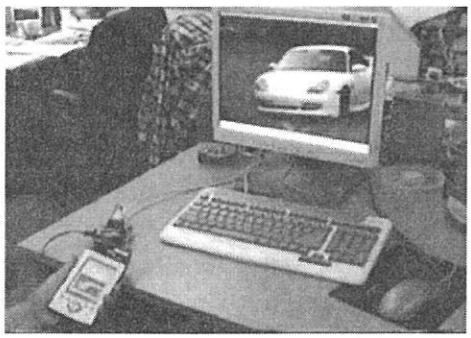

Fig. 5. Smart view implementation

4.3 MBASE 적용결과

스마트 뷰 구현을 통하여 찾아낸 이슈(issue)들로는 이질적인 개발 플랫폼, 어플리케이션의 상호운용성, 서비스 컴포넌트 할당 및 분배를 위한 유비쿼터스 컴퓨팅 아키텍쳐, 소프트웨어 디버깅 및 검증 문제 등이다[7]. 이러한 이슈들은 일반적인 유비쿼터스 컴퓨팅 시스템 특징인 정보기기에 컴퓨팅과 통신기능의 추가로 인한 이해관계자들의 다양화, 컨버전스(convergence), 정보기기의 저가격화, 상황에 따른 제공 서비스의 변화 등으로부터 기인한 것이며, 또한 임베디드 시스템의 일반적인 하드웨어 제약사항, 시스템 비용과 같은 특수한 제약사항과 관련된 문제점이다.

Table 2는 스마트 뷰 개발시 발생되는 모델간의 충돌을 해결하기 위하여 MBASE 방법론에서 제시하고 있는 방안들을 적용한 결과이다.

Table 2. Results of applying the MBASE

구 분	유비쿼터스 이슈	MBASE 적용결과
Product-Property-Success	이기종(heterogeneous)의 개발 플랫폼 간의 부조화로 인한 문제점 발생	이기종간의 변화 분석 기법을 적용하여 동일 한 기능을 제거 하거나 두 시스템 간에 누락된 기능을 부여함으로써 해결 함
Product-Property	서비스 컴포넌트 할당 및 분배를 위한 유비쿼터스 아키텍처가 필요함	시스템엔지니어링 프로세스의 기능분석 및 할당과 유사한 MBASE 프로세서의 LCO(Life Cycle Architecture) 단계를 거치면서 해결됨
Product-Success	서로 다른 이질적인 시스템간의 상호 운용성 문제	COTS, legacy 등의 활용에 적합한 객체모델기반 개발방법론의 적용으로 문제 해결 가능
Process-Property	소프트웨어 디버깅 및 검증 상의 문제점 발생	복합시스템 개발방법과 하드웨어와 소프트웨어 Co-design 개발방법을 적용하여 절충방법 등으로 하드웨어에 적합한 소프트웨어 개발가능

5. 결론

본 논문에서는 MBSE 방법을 적용하여 유비쿼터스 컴퓨팅 시스템 분야에 특성화된 개발 방법론을 제시하였다. 유비쿼터스 컴퓨팅 시스템 개발 시 발생할 수 있는 모델들 간의 충돌을 찾아내고 이를 방지하기 위하여 요구사항 단계를 포함한 각 단계에서의 개발기술을 MBASE 접근방법을 기반으로 제시하였다. 연구결과 MASE 접근법은 유비쿼터스 컴퓨팅 시스템에 일반적인 방법론 적용시 발생 가능한 몇 가지 문제점을 해결할 수 있다는 결론을 얻었다.

본 논문에서 제시한 개발방법을 유비쿼터스 컴퓨팅 도메인에 일반화 할 수 있는지를 검증하

기 위하여 몇 가지 프로젝트에 적용하였지만 평가나 검증 결과로 제시하기에는 미흡한 점이 있었다. 향후에 MBASE 또는 개선된 개발방법론을 유비쿼터스 컴퓨팅 도메인에서 일반적으로 적용할 수 있도록 하기 위하여 유비쿼터스 컴퓨팅 시스템에 적합한 버전 개발을 위한 연구를 수행할 계획이다.

참고문헌

1. Boehm, B. and Port D., "Escaping the Software Tar Pit: Model Clashes and How to Avoid Them," Software Engineering Notes, Association for Computing Machinery, 1999.
2. http://sunset.usc.edu/research/MBASE.
3. Boehm, B., Port, D., Alsaid, M., "Avoiding the Software Model-Clash Spiderweb", IEEE Software, November, 2000.
4. Boehm, B., Hoh. In, "Identifying Quality-Requirement Conflicts". IEEE Software, March 1996.
5. Hoh In, C. Kim, and S. Yau, "Q-MAR:An Adaptive Qos Management Model for Situation -Aware Middleware," EUC 2004, pp. 972-981, 2004.
6. J. Han, Hoh In, J. Woo, "Towards Situation -aware Cross-platform Ubi-Game Development" 11th IEEE Asia-Pacific Software Engineering Conference 2004, pp 734-735, 2004.
7. Hoh. In, Dong-hyun Lee, "Software Engineering Issue for Ubiquitous Entertainments Service", 7th SEKE (Software Engineering & Knowledge Engineering) Conference 2005, 2005.