커뮤니티 컴퓨팅 어플리케이션 개발을 지원하기 위한 이클립스 기반의 통합개발환경

김동욱°, 이정태, 류기열
아주대학교 정보통신전문대학원 정보통신공학과
{neodende°, jungtae, kryu}@ajou.ac.kr

IDE(Integrated Development Environment) based on Eclipse for Community Computing Application Development

Dongwook Kim, Jungtae Lee, Kiyeol Ryu
Dept. of Information and Communication Engineering, Ajou University

요 약

커뮤니티 메타포를 이용하여 유비쿼터스 공간을 정의하고 유비쿼터스 서비스를 구현하는 커뮤니티 컴퓨팅에 대한 연구가 제안되고 있다. 제시된 커뮤니티 컴퓨팅 어플리케이션의 개발과정은 추상화된 하이레벨의 커뮤니티 컴퓨팅 모델로부터 최종 플랫폼에 적합한 코드를 생산하는 MDA 개발방법이 적용되었다. 그러나 이러한 개발과정을 효율적으로 지원하는 통합개발환경은 아직 존재하지 않는다. 최근 관심이 고조되고 있는 이클립스는 오픈소스 프로젝트, 공개표준 런타임, 다양한 기능을 제공하는 플러그인들의 지원 등의 장점을 가지고 있기 때문에 통합개발환경을 구성하기 위한 적절한 방안으로 간주된다. 따라서 본 논문에서는 커뮤니티 컴퓨팅의 개발과정을 지원할 수 있는 통합개발환경을 이클립스를 기반으로 하여 연구하였다.

1. 서 론

유비쿼터스 시스템을 구축하기 위한 패러다임으로써 커뮤니티 메타포를 이용하는 커뮤니티 컴퓨팅(Community Computing)이 제안되었다. 커뮤니티 컴퓨팅이란 유비쿼터스 공간(U-space)내에서 존재하는 각종 엔티티(Entity)들이 커뮤니티를 형성하고, 엔티티들 간의 상호협력을 통하여 유비쿼터스 서비스를 제공하는 것이다. 현재 커뮤니티 컴퓨팅을 실현하기 위한 모델로써 제시된 멀티 에이전트 기반의 커뮤니티 컴퓨팅 시스템[1]은 MDA 개발방법[2]을 적용하여 하이레벨의 추상화된 모델로부터 모델 변환과정을 거쳐서 최종코드를 생산한다.

다양한 통합개발환경들이 있지만 커뮤니티 컴퓨팅을 위한 통합개발환경은 제시되어 있지 않기 때문에 새로운 통합개발환경이 필요하다. 최근 관심이 고조되고 있는 이클립스의 리치클라이언트 플랫폼은 다음과 같은 장점을 가지고 있다. 리치클라이언트 플랫폼에서 개발된 GUI 어플리케이션은 이클립스가 동작하는 모든 플랫폼에서 사용가능하고, 다양한 플러그인을 자신의 어플리케이션에서 활용할 수 있으며, 개발이 완료된 어플리케이션을 실행가능한 배포본으로 만들 수 있다.[3][4] 따라서, 이클립스의 리치클라이언트 플랫폼에 커뮤니티 컴퓨팅의 구현에 필요한 기능들을 플러그인으로 통합하고, 이에 대한 사용자 인터페이스를 제공함으로써 커뮤니티 컴퓨팅을 위한 통합개발환경이 가능하다.

이를 위하여 본 논문에서는 커뮤니티 컴퓨팅의 사례로써 연구된 일련의 개발과정을 토대로 커뮤니티 컴퓨팅 어플리케이션 개발자에게 공통적으로 요구되는 통합개발환경의 기능을 분석하고, 이클립스를 기반으로 그 기능을 지원하기 위한 방안에 대해서 논의함으로써 커뮤니티 컴퓨팅 어플리케이션 개발에 적합한 통합개발환경의 형태를 제안한다. 본 논문의 구성은 다음과 같다. 2장에서 커뮤니티 컴퓨팅 어플리케이션의 개발과정을 분석하고, 3장에서 이클립스의 리치클라이언트 폼에 대하여 기술한 후에, 4장에서 이클립스를 기반으로 개발되는 통합개발환경을 제안한다. 마지막으로 5장에서 결론 및 향후 연구방향에 대하여 제시한다.

2. 커뮤니티 컴퓨팅 어플리케이션의 개발과정

커뮤니티 컴퓨팅 어플리케이션 개발에서 선행되어야 하는 과정은 커뮤니티 모델링 언어(CML:Community computing Modeling Language)를 사용하여 커뮤니티 컴퓨팅 모델(CCM:Community Computing Model)을 기술하는 것이다. 이 모델에는 유비쿼터스 공간의 정의와 커뮤니티의 타입(목표, 역할, 상호협력을 위한 규약등)이 기술되어야 한다. 유비쿼터스 공간에는 멤버들이 존재하는데 이는 컴퓨팅 디바이스, 소프트웨어, 사람등으로써 에이전트가 될 수 있는 자원이다. 모델로부터 구현된 유비쿼터스 공간이 동작하는 과정은 다음과 같다. 유비쿼터스 공간 내에서 목표(Goal)가 발생하면 목표를 위한 커뮤니티 타입의 역할과 그 역할을 수행할 수 있는 멤버를 동적으로 바인딩한다. 이후에 멤버들 간의 협업과정을 통하여 목표(유비쿼터스 서비스)를 달성한다.

CCM에서는 커뮤니티의 수행에 필요한 유비쿼터스 공간의 요구사항을 표현하고 있는 반면에 CIM-PI(Platform Independent Community Computing Implementation Model)는 그 요구사항을 특정 플랫폼과 무관한 형태로

구현한 구현 수준의 모델이다. CIM-PI에서는 CCM에 기술되지 않은 멤버들 간의 통신, 즉 목표를 이루기 위한 멤버들 간의 메시지 교환을 SEQ, PAR, ALT, CASE, WHILE, EXIT등의 구문을 이용하여 기술한다.

CIM-PI로부터 CIM-PS(Platform Specific Community Computing Implementation Model)로 변환하는 과정은 특정 플랫폼에 무관한 모델에서 특정 플랫폼에 대한 코드를 생산하는 것이다. 그림 1은 CCM으로부터 CIM-PS로 변환하는 전체 개발과정을 도식화한 것이다.

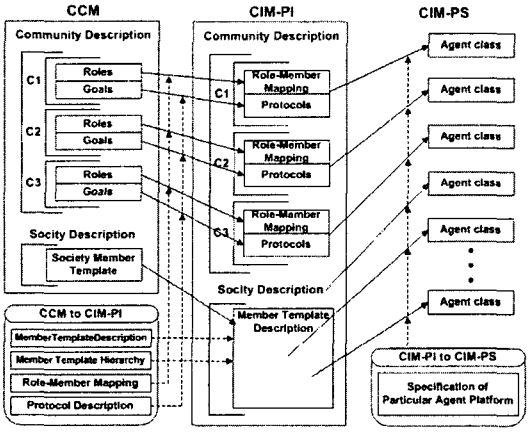

그림 1. CCM으로부터 CIM-PS로의 변환과정

3. 이클립스의 리치클라이언트 플랫폼

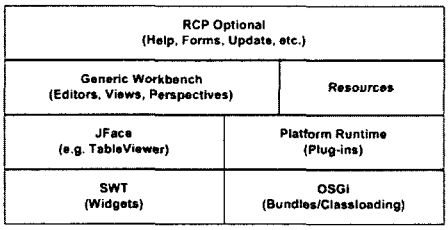

그림 2. 리치클라이언트 플랫폼

이클립스의 리치클라이언트 플랫폼(RCP:Rich Client Platform)은 리치클라이언트 어플리케이션을 개발할 때 필요한 상호의존적인 플러그인들을 모아놓은 이클립스 플랫폼의 서브셋이다. 즉, 이클립스 플랫폼에서 지원하는 기능들을 사용하는 것이 가능하고, 플러그인들을 추가하여 확장하는 것이 가능하며, Generic Workbench와 SWT를 사용하여 리치클라이언트 플랫폼을 사용하여 어플리케이션을 개발하는 개발자가 임의로 사용자 인터페이스를 구성하는 것이 가능하다.[5][6]

리치클라이언트 플랫폼으로 개발된 어플리케이션은 새로운 통합개발환경이 될 수 있으며, 이러한 어플리케이션은 OSGi 런타임에 기반하기 때문에 플러그인들을 결합하여 계속적인 확장이 가능한 장점이 있다. 리치클라이언트 플랫폼은 운영체제에 관계없이 어플리케이션 개발이 가능하고, 배포한 어플리케이션은 동적인 업데이트가 가능하기 때문에 추후에 유지 및 보수가 용이하다.[7]

4. 통합개발환경

현재 커뮤니티 컴퓨팅 모델을 정의하는 방법은 커뮤니티 모델링 언어를 사용하여 텍스트 기반으로 모델을 기술하는 것이다. 본 논문의 통합개발환경에서는 텍스트 기반이 아닌 그래피컬한 모델링 에디터를 제공한다. 그래피컬하게 모델을 기술하면 커뮤니티, 멤버, 멤버의 역할, 커뮤니티간의 관계등이 쉽게 파악된다. 또한 텍스트로 모델을 기술할 때 반복되는 코드작성을 감소시키고, 커뮤니티 모델링 언어 문법을 따르지 않는 코드의 작성을 사전에 차단함으로써 잘 정의된 모델의 기술이 가능하다. 통합개발환경에서 제공하는 그래피컬한 모델링 에디터는 다음과 같은 방법으로 모델을 기술한다. 먼저 유비쿼터스 공간을 정의하고, 다음으로 멤버들을 유비쿼터스 공간으로 위치시키고, 마지막으로 멤버의 인스턴스와 커뮤니티의 역할을 맵핑시킨다. 그림 3은 이러한 과정을 도식화한 것이다.

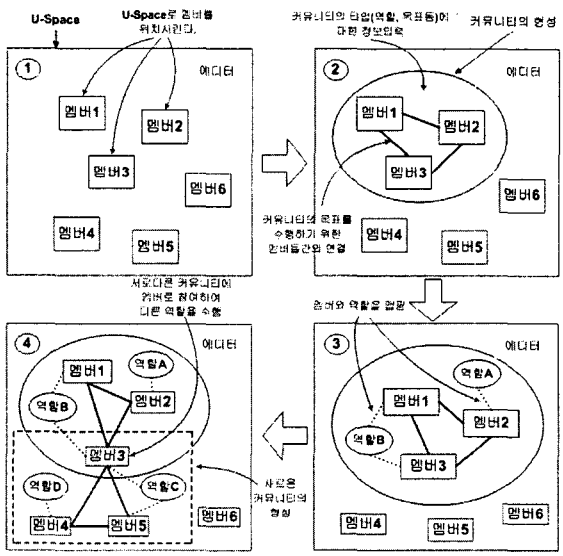

그림 3. 그래피컬한 커뮤니티 컴퓨팅 모델의 기술

위와 같은 모델링에서 얻어지는 정보를 바탕으로 자동으로 스크립트 형태로 변환하여, 통합개발환경을 사용하는 개발자가 모델 기술 이후의 변환과정을 순차적으로 수행할 수 있게 한다. 정의한 모델은 필요시 수정이나 확장이 가능해야 한다. 이를 위하여 본 논문의 통합개발환경에서는 그래피컬한 모델 또는 텍스트 기반의 모델에서 선택적으로 수정작업이 가능하도록 지원한다.

본 논문의 통합개발환경은 이클립스의 GEF(Graphical Editing Framework)[8]를 활용하였다. GEF는 그래피컬한 에디터의 개발에 필요한 플러그인들을 지원한다. 이

를 이용하여 커뮤니티 컴퓨팅의 구성요소(멤버, 역할, 멤버-역할 맵핑등)를 드래그 소스로 제공하는 팔레트를 구성한다. 통합개발환경의 사용자는 마우스로 드래그소스를 드래그하여 다이어그램에 표현한다. 또한 드래그 소스로 제공할 수 없는 것은 뷰를 통하여 입력받는다. 이러한 작업을 통하여 커뮤니티 컴퓨팅 모델을 기술한다.

커뮤니티 모델의 개발을 완료한 후에는 정의한 모델에 대한 실제 코드를 생산할 수 있어야 한다. 이를 위하여 변환과정을 개발하는 개발자는 CCM을 CIM-PI로, CIM-PI를 CIM-PS로 변환하는 과정을 각각 플러그인으로 개발해야 한다. 이때 각 변환과정에서 요구되는 기능이 서로 다르기 때문에 이를 효율적으로 지원하기 위하여 변환과정에 따라 통합개발환경의 퍼스펙티브를 다르게 제공할 필요가 있다. 이클립스의 리치클라이언트 플랫폼은 퍼스펙티브, 메뉴바나 툴바의 내용등을 컨트롤하는 워크벤치 어드바이저를 제공하고, 뷰와 에디터의 배치를 커스터마이즈 할 수 있다. 이를 이용하여 본 논문의 통합개발환경에서는 기존 이클립스가 가지고 있는 여러 가지 사용자 인터페이스 중에서 각 단계별로 필수적인 요소들로 구성된 사용자 인터페이스를 제공한다. 그림 4는 통합개발환경에서 요구되는 플러그인들을 결합하는 방법을 도식화한 것이다.

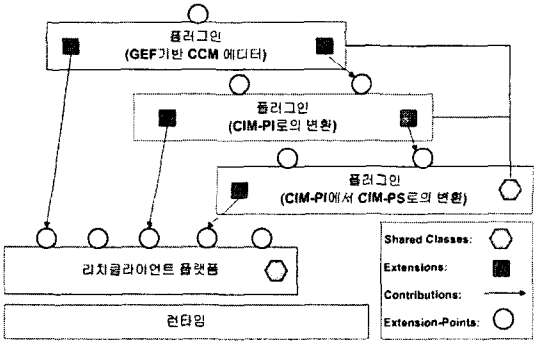

그림 4. 플러그인과 플랫폼의 결합

본 논문의 통합개발환경은 다양한 플랫폼 종속적인 코드로의 변환을 지원하는 기능을 제공한다. 동일하게 기술된 모델을 이용하여 JADE 기반의 멀티 에이전트 플랫폼에서 동작하도록 하는 것 이외에도 EJB와 같은 컴포넌트 기반 플랫폼, 또는 CORBA와 같은 분산 어브젝트 기반 플랫폼에서 동작하는 코드를 생산해 내는 기능을 지원해야 한다. 이러한 기능을 지원하는 것은 각각의 변환과정을 플러그인으로 개발하는 통합개발환경 개발자의 몫이다. 그러나 통합개발환경을 사용하는 사용자가 많은 플러그인 중에서 어떤 플러그인을 선택할지를 결정하는 것은 통합개발환경에서 제공해야 한다. 이를 위하여 본 논문의 통합개발환경은 플랫폼을 선택할 수 있게 하는 사용자 인터페이스를 제공하고, 이에 따라 변환과정에 필요한 플러그인으로 교체하는 기능을 제공한다. 그림 5는 커뮤니티 컴퓨팅 개발과정에 대응되는 본 논문의 통합개발환경에서의 지원방안을 도식화한 것이다.

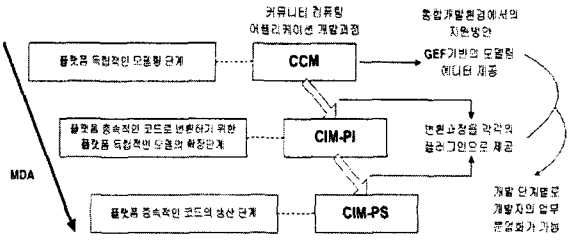

그림 5. 개발과정에 대한 통합개발환경의 지원방안

5. 결론 및 향후 연구 방향

본 논문에서는 커뮤니티 컴퓨팅 환경을 지원하는 어플리케이션 개발을 수행하는 데에 필요한 요구사항을 파악하여 커뮤니티 컴퓨팅 어플리케이션 개발자에게 개발시간의 단축, 개발업무의 분업화, 개발과정의 연계성을 통한 일관성 있는 개발과 같은 편의를 제공하기 위해서 필요한 기능들을 지원하는 통합개발환경을 제안하였다. 제안한 통합개발환경은 이클립스의 리치클라이언트 플랫폼을 기반으로 하였기 때문에 추가로 요구되는 플러그인을 통한 확장이 가능하고 배포본으로 만들어서 배포가 가능하며 업데이트 기능을 이용하여 계속적인 유지, 보수가 가능하다.

현재 본 논문에서 제안한 통합개발환경의 구현이 진행중이며, 다양한 플랫폼을 위한 변환도 지원할 수 있도록 추가적인 플러그인의 개발에 대한 연구도 진행되고 있다.

참고문헌

[1] Yuna Jung, Jungtae Lee, Minkoo Kim, "Multi-agent based Community Computing System Development with the Model Driven Architecture", AAMAS, 2006.

[2] OMG, Model Driven Architecture, "MDA Guide Version 1.0.1", http://www.omg.org/mda, June 2003.

[3] Eclipse.org, "Eclipse Platform Technical Overview", http://www.eclipse.org/articles/Whitepaper-Platform-3.1/eclipse-overview-2005-12.pdf, 2005.

[4] Jim D'Anjou, Scott Fairbrother et al, "The Java Developer's Guide to Eclipse-Second Edition", 2004.

[5] Eclipse.org, "Notes on the Eclipse Plug-in Architecture", http://www.eclipse.org/articles/Article-Plug-in-architecture/plugin_architecture.html, 2003.

[6] Eclipse.org, "Eclipse Forms: Rich UI for the Rich Client", http://www.eclipse.org/articles/Article-Forms/article.html, 2005.

[7] Eclipse.org, "Dev guide: Updating a product or extension", http://help.eclipse.org/help31/index.jsp?topic=/org.eclipse.platform.doc.isv/guide/product_update.htm, 2006.

[8] IBM, "Eclipse Development-using the Graphical Editing Framework and the Eclipse Modeling Framework", ibm.com/redbooks, 2004.