커뮤니티 컴퓨팅: 협업 기반 환경 자동 적응의 컴퓨팅 모델†

아주대학교 강경란* · 김민구** · 이정태**
정유나*** · 조위덕** · 길현숙

1. 개 요

컴퓨팅 기술과 네트워킹 기술이 발전하면서 이를 활용하여 보다 지능적이고 자연스러운 서비스를 제공하는 유비쿼터스 컴퓨팅 환경을 개발하는 노력들이 다양하게 진행되고 있다.

이러한 노력들은 분산 객체 기술, 다중 에이전트 기술, 분산 협력 기술, 웹 기반 서비스 아키텍처 등으로 예시되는 기존의 분산 컴퓨팅 기술들 중에서 자신들이 추구하는 서비스 모델에 적합한 컴퓨팅 모델을 선택하여 유비쿼터스 컴퓨팅 환경을 개발하고 있다[1-10]. 예를 들어, Aura [1]나 GAIA [2] 같은 경우에는 분산 객체 기술을 활용하여 사용자의 이동에 상관없이 그리고 컴퓨팅 환경의 다양성에 무관하게 서비스가 지속되도록 하는 아키텍처를 제시하고 있다. HP Cooltown [3]의 경우에는 이동 사용자와 환경 간의 매끄럽고 자연스러운 상호 작용을 달성하기 위해 웹 기반의 서비스 아키텍처를 활용하고 있다.

본 연구에서는 협력적이고 적응적인 서비스 개발을 목표로 하여, 이러한 서비스들을 보다 자연스럽고 직관적으로 개발할 수 있도록 하는 새로운 컴퓨팅 모델인 '커뮤니티 컴퓨팅(Community Computing)'[11]을 제시한다. '커뮤니티'라는 용어는 새로운 것이 아니고 동일 지역에 사는 사람들이나 동일한 취미를 가진 사람들의 집합이라는 개념으로 이미 널리 사용되어 오고 있다[12-16]. 이러한 개념의 연장선 상에서, 커뮤니티 컴퓨팅에서는 하나의 서비스를 다수의 협력하는 개체들의 집합인 '커뮤니티'로 모델링하고, 커뮤니티에 참여하는 개체들 사이의 상호 작용을 통해 서비스가 완성된다. 커뮤니티가 구성되는 조건의 역동성에 따라 커뮤니티의 종류를 크게 정적(static), 동적(dynamic), 자율적(autonomic) 커뮤니티로 구분한다.

커뮤니티 컴퓨팅 개념을 구체적인 서비스로 실체화하기 위한 방안으로써, 두 가지 접근 방법으로 연구를 진행하고 있다. 첫번째 방법으로, 장기적인 개발 방법론으로서, Model Driven Architecture[17]에 근거한 커뮤니티 시스템 개발 방법론[18]을 연구하고 있다. 응용 서비스 개발자로 하여금 목표로 하는 유비쿼터스 컴퓨팅 공간과 이 공간에 존재하는 개체들 사이의 상호 작용을 추상적이고 논리적인 형식으로 모델링하고 이 모델을 기반으로 실제 동작 가능한 프로그램을 추출해 내는 방법론이다. 두번째 방법은, 단기적인 개발 방법으로서, 기존의 서비스를 커뮤니티의 구성원으로 참여시키기 위한 방법론[19]이다. 특수한 목적의 언어를 개발하여 이 언어를 사용해서 커뮤니티를 기술하고, 중앙의 '커뮤니티 매니저'라는 서버가 존재해서 기술된 언어를 해석해서 커뮤니티를 구성하고 관리하는 역할을 수행하는 방법론이다.

본 고의 구성은 다음과 같다. 2절에서 커뮤니티 컴퓨팅에 대한 기본적인 개념을 소개할 것이고, 3절에서 제안하는 커뮤니티 모델을 보다 자세하게 설명하고 커뮤니티 컴퓨팅 개념을 구현하기 위한 방법론을 4절에서 설명할 것이다. 5절에서 본 고를 마무리하며 앞으로의 연구 목표와 방향에 대해 기술할 것이다.

2. 커뮤니티 컴퓨팅의 개념

2.1 기본 용어와 개념

본 연구에서는, 유비쿼터스 컴퓨팅 공간을 'uT-공간(uT-space)'라고 부르고, "서로 상이하고 (경우에 따라서는 이동성을 가진) 컴퓨팅 개체들이 동적으로 조정되고 통신이 가능한 환경"으로 정의한다. 하나의 uT-공간이 하나의 물리적인 공간에 대응될 수도 있지만, 이동성이 있는 개체들이 다른 물리적인 공간으로 이동하더라도 통신이 가능하므로 물리적인 경계는 제

† 본 연구는 21세기 프론티어 연구개발사업의 일환으로 추진되고 있는 정보통신부의 유비쿼터스 컴퓨팅 및 네트워크 원천 기반 기술 개발 사업의 지원에 의한 것임.
* 정 회 원
** 종신회원
*** 학생회원

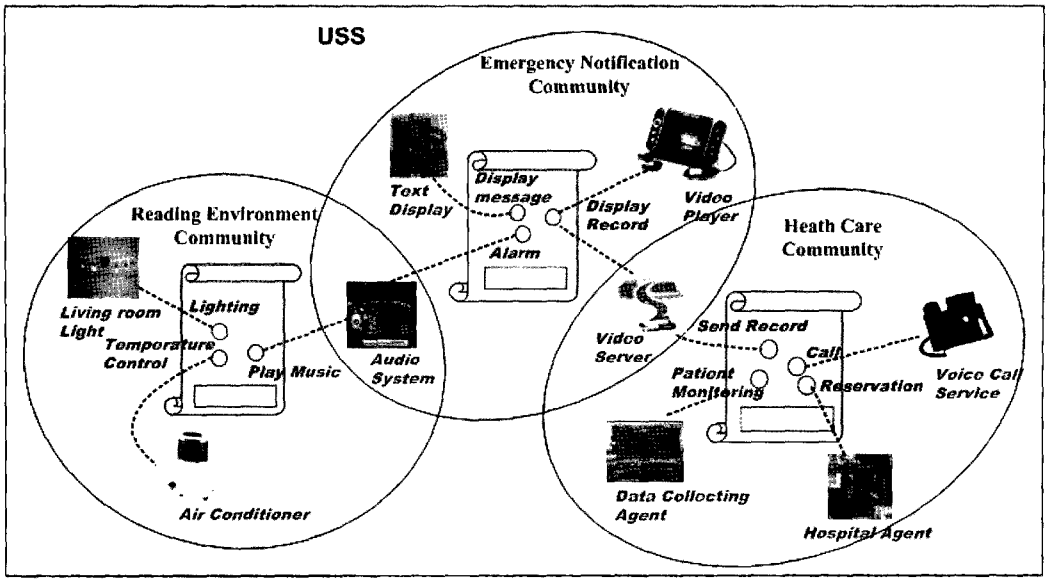


그림 1 USS의 uT-개체들과 커뮤니티 실례의 관계 사례

한적이지 않고 확장 가능하다고 가정한다.

uT-공간 내에 존재하는 컴퓨팅 개체를 'uT-개체(uT-entity)'라 부른다 uT-개체에는 시스템 프로세스나 응용 프로그램과 같이 다양한 종류의 소프트웨어가 포함되고, 컴퓨터, 센서, 디지털 가전 등과 같이 다양한 하드웨어들도 포함된다. 사람도 uT-개체들을 사용하고 uT-개체들과 상호 작용하므로 uT-공간의 구성요소라고 할 수 있다.

본 연구에서 추구하는 uT-공간은 사람들의 웰빙을 도모하는 것을 목적으로 하는 '유비쿼터스 지능 공간(Ubiquitous Smart Space, USS)'이다. USS 내에서 동적으로 변화하고 복잡한 서비스를 제공하기 위해서는 uT-개체들이 서로 협력해야 한다. 이렇게 공통된 서비스 목적을 가지고 협력하는 uT-개체들의 집합을 커뮤니티라 한다. 즉, 커뮤니티는 협력하는 개체들의 집합에 대한 상징이다. '커뮤니티 컴퓨팅'은 커뮤니티라는 상징을 사용해서 서비스를 개발하는 컴퓨팅 모델을 의미한다. 커뮤니티를 통해 제공되는 서비스는 '목적(goal)'을 가지며, 참여하는 uT-개체들이 수행해야 하는 '역할(role)'과 참여하는 개체들 간에 협력하는 과정을 수행하는 과정을 정의한 '글루(glue)'를 갖는다. 이러한 개별 커뮤니티의 집합된 정보를 '커뮤니티 템플릿(community template)'이라 한다. 커뮤니티 템플릿에 근거해서 목적을 달성하기 위해 현재 모여 있는 uT-개체들의 집합을 '커뮤니티 실례(community instance)'라 한다.

특정 역할을 수행할 uT-개체를 선택해서 커뮤니티에 참여하는 과정을 '역할-구성원 바인딩(role-member binding)'이라 하고, 해당 uT-개체는 커뮤니티 실례에 '가입한다(join)'고 말한다. 역할을 수행하기 위해서는 글루에 따라 절차를 진행해야 하는데, 커뮤니티 컴퓨팅의 궁극적인 목표는 이러한 역할-구성원 바인딩이나 글루의 결정이 상황에 따라 자율적으로 이루어지도록 하는 것이다.

그림 1은 USS와 커뮤니티 실례에 대한 예를 보이고 있다. 그림 1에서 보이는 바와 같이, 하나의 USS 내에 여러 개의 커뮤니티들이 존재할 수 있으며, 하나의 uT-개체가 여러 커뮤니티 실례에 참여하는 것이 가능하다. 참여하는 커뮤니티에 따라 글루가 달라지므로 동일한 uT-개체의 동작 (behavior)가 달라질 수 있다.

커뮤니티 컴퓨팅에 근거해서 응용 서비스 프로그램을 개발할 때는, 현재의 필요에 따라 우선 개발을 하고, 추후에 필요하게 되는 서비스에 대해서는 커뮤니티 템플릿의 형식으로 필요한 기능과 상호 작용을 정의한다. 템플릿에 부합하는 기존 응용 프로그램들을 선택하여 커뮤니티 실례를 구성함으로써 목표로 하는 서비스가 제공될 수 있도록 한다. 다만, 각 uT-개체는 커뮤니티에서의 특정한 역할을 수행하기 위한 글루를 설치해서 동작할 수 있는 준비가 되어 있어야 한다.

커뮤니티 컴퓨팅이 일반 사용자를 대상으로 하는 서비스 분야 외에도 네트워크 분야 등과 같이 다양한 분야에 적용될 수 있다. 센서 네트워크의 경우를 예로 들자면, 현재의 감지 요구를 만족시키기 위해 센서와 싱크, 액터가 설치될 것이다. 이렇게 설치된 센서 네트워크들과 싱크, 액터들을 결합하여 보다 발전된 형태의 감지 서비스를 커뮤니티의 형식으로 구현할 수 있다.

네트워크 서비스에 있어서도, 라우터들이 보다 지능적인 처리가 가능하다면, 현재의 필요에 따라 라우터를 설치하고 이들을 커뮤니티의 형식으로 결합하여 다양한 QoS(Quality of 서비스) 및 다중 전달 경로 서비스 등과 같이 보다 발전된 형태의 서비스를 제공할 수 있다.

2.2 특징과 장점

기존의 분산 객체 모델[20]이나 분산 협동 모델[20]은 원격에 있는 객체들의 서비스 인터페이스를 활용하여 서비스를 개발하게 함으로써 서비스 시스템 외부에 있는 다른 컴퓨팅 자원을 활용할 수 있게 한다. 가용 원격 서비스를 검색하고 연결시켜 주는 기능을 담당하기 위한 오브젝트 브로커나 공유 데이터 공간 등이 있어서 응용 프로그램의 관점에서는 컴퓨팅 환경의 다양성이나 위치에 별다른 제약을 갖지 않고 객체나 서비스 인터페이스에 접근할 수 있다. 그러나, 해당 서비스에 필요한 원격 객체의 서비스 인터페이스를 호출하여 사용하고 서비스를 완성해야 하는 응용 프로그램이 존재해야 한다. 즉, 서비스를 위해 이를 목적으로 하는 응용 프로그램이 개발되어야 한다.

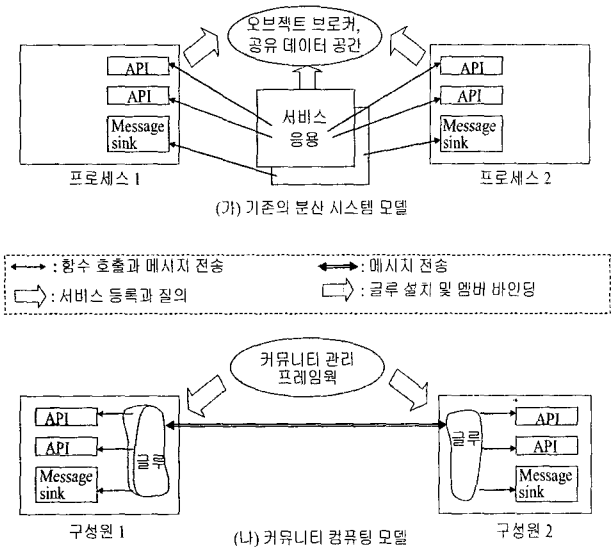


그림 2 기존의 분산 시스템 모델과 제안하는 커뮤니티 컴퓨팅과의 비교

커뮤니티 컴퓨팅에서는 이러한 목적을 위해 개별 응용 프로그램을 개발하는 것이 아니라, 커뮤니티 템플릿을 개발하는 것으로 필요한 서비스를 정의할 수 하다. 그림 2에서 보이는 바와 같이, 커뮤니티 관리 프레임웍에서 커뮤니티 템플릿을 관리하고 역할-구성원 바인딩을 수행한다. 필요에 따라 새로운 커뮤니티 템플릿을 커뮤니티 관리 프레임웍에 설치하고, 커뮤니티 템플릿이 제공하는 글루를 기존의 응용 프로그램에 설치함으로써 해당 커뮤니티 가 추구하는 서비스를 제공할 수 있다.

서비스가 동작하는 상황의 관점에서 보면, 기존의 분산 모델은 중앙의 서비스 프로그램이 필요한 서비스 인터페이스를 일일이 호출해서 사용함으로써 서비스가 동작하지만, 커뮤니티 컴퓨팅을 적용하면 개별 참여 uT-개체가 글루에 따라 자율적으로 동작하므로 일상생활에서의 협력하는 모델을 보다 자연스럽게 개발하는 것이 가능하다.

두번째, 커뮤니티 컴퓨팅은 기존의 uT-개체를 보다 다양하게 활용할 수 있게 한다. 커뮤니티의 구성원로서 동작하기에 필요한 글루를 설치하면 해당 커뮤니티에서의 역할을 수행할 수 있다. 즉, 하나의 uT-개체를 커뮤니티에 따라 다른 용도로 역할을 담당하도록 할 수 있다. 기존의 분산 모델의 경우에도 기존의 uT-개체의 서비스를 다양하게 활용할 수 있지만, 이미 외부에 공개된 인터페이스에 대해서만 가능하고, 프로그램 내의 내부 인터페이스는 외부에서 접근할 수가 없다. 그러나, 커뮤니티 컴퓨팅에서는 글루가 uT-개체 내에 설치되어 동작하는 것이므로 외부에 공개되지 않은 인터페이스도 활용하는 것이 가능하다.

세번째, 커뮤니티 컴퓨팅을 적용하면 서비스 제공자로 하여금 기존의 커뮤니티 템플릿을 변경하는 것으로 새로운 서비스를 개발할 수 있게 한다. 커뮤니티 템플릿은, 앞서 설명한 바와 같이 목적, 역할, 글루 등과 같이 변경 가능한 구성요소들로 구성되어 있으므로, USS 내의 uT-개체 구성이 변경되거나 서비스에 대한 수요에 변경이 있을 때, 기존의 템플릿을 변경하는 것만으로도 변화된 환경에 적응하는 새로운 서비스 제공이 가능하다. 기존의 분산 시스템 모델에서는 새로운 수요를 만족시키기 위해서는 이에 맞는 새로운 응용 프로그램을 개발해야 한다.

3. 커뮤니티 모델

앞서 설명한 바와 같이, 커뮤니티는 협동적인 서비스의 상징이다. 협동적인 서비스를 개발하는데 있어, 커뮤니티 컴퓨팅은 uT-개체들의 집합이라는 관점에서 커뮤니티 템플릿이라는 것을 사용해서 서비스를 기술한다. 이 커뮤니티 템플릿의 역동성에 따라 커뮤니티의 종류를 정적 커뮤니티, 동적 커뮤니티, 자율적 커뮤니티로 구분할 수 있다.

3.1 커뮤니티의 종류

3.1.1 정적인 커뮤니티

정적인 커뮤니티는 커뮤니티의 초보적인 모델이다. 커뮤니티 템플릿은 구성원으로서 참여해야 할 구체적

인 개체들을 기술하고 있고, 글루도 고정적이고, 전혀 가변적인 요소를 갖고 있지 못하다. 다시 말해서, 역할-구성원 바인딩이 템플릿이 적용되는 순간에 완결된다. 그러므로, 정적인 커뮤니티의 실례는 해체되기 전까지 USS 내의 uT-개체의 변화나 사용자의 이동에 따른 환경의 변화에도 불구하고 역할-구성원 바인딩이 변화되지 않는다. 이러한 형태의 커뮤니티 종류는 커뮤니티 컴퓨팅에 의한 서비스 개발이 시작되는 시점에서는 일부 있을 수 있겠지만, 좀 더 기술이 성숙되면 자연적으로 감소될 것이다.

3.1.2 동적인 커뮤니티

동적인 커뮤니티는 정적인 커뮤니티에 비해 보다 발전된 형태의 커뮤니티다. 커뮤니티 템플릿 자체는 고정적이지만, 템플릿이 사용되는 시점, 장소, 사용자 요구사항에 따라 역할에 연결되는 구성원이 달라질 수 있다. 커뮤니티 실례가 생성되는 시점에 역할-구성원 바인딩이 이루어지기 때문에 이런 변화가 가능하다. 그리고, 이미 생성된 커뮤니티 실례에 대해서도 사용자가 이동해 가면 위치적인 요구사항을 갖는 역할에 대해서는 요구 사항을 만족시키기 위해 해당 uT-개체가 새롭게 선택된다.

3.1.3 자율적인 커뮤니티

자율적인 커뮤니티는 가장 높은 역동성을 갖는 커뮤니티다. 커뮤니티 템플릿도 커뮤니티의 필요성이 제기될 때 만들어지고 USS 내의 uT-개체 변화에 따라 커뮤니티 템플릿이 적응적으로 변화한다. 해당 서비스를 위한 커뮤니티 템플릿을 미리 생성해 두지 않아도, 커뮤니티에 대한 필요가 발생하면, 커뮤니티 관리 프레임웍의 도움으로 uT-개체의 요청에 의해 커뮤니티 템플릿이 생성될 수 있다. 기존의 커뮤니티 템플릿을 일부 수정하는 것으로 새로운 커뮤니티를 위한 템플릿을 만들어 낼 수도 있고, 전혀 새로운 것을 생성하는 것도 가능하다. 이러한 종류의 커뮤니티가 커뮤니티 컴퓨팅에서 추구하는 궁극적인 바이다.

3.2 커뮤니티의 생애 모델

개념적으로 커뮤니티 실례가 생성에서부터 삭제에 이르는 생애를 갖는다. 각 생애 단계의 특징에 대해서 기술하면 아래와 같다.

3.2.1 생성(creation)

커뮤니티 실례는 커뮤니티 관리 프레임웍이거나 일반적인 uT-개체에 의해서 생성될 수 있다. 커뮤니티가 제공하는 서비스를 요청하는 uT-개체에 의해 해당 서비스를 위한 커뮤니티 실례가 생성될 것이고, USS의 컨텍스트에 따라 관리 프레임웍에 의해 필요한 커뮤니티 실례가 생성될 수 있다. 커뮤니티 실례 생성에 있어서 제일 중요한 작업은 역할-구성원 바인딩이다.

3.2.2 활성화(activation)

커뮤니티 실례는 생성되는 순간 바로 활성화될 수 있다. 활성화된다는 것은 구성원 uT-개체들이 글루를 설치하고 이 글루에 따라 타 구성원들과 상호작용하는 것을 의미한다. 커뮤니티 실례는 제공하는 서비스의 종류에 따라 그 목적을 달성한 후에 삭제되는 것이 아니라, 비활성화와 활성화를 반복할 수 있다. 동적인 커뮤니티의 경우에는, 재활성화되는 시점, 위치, 사용자의 요구의 변화 등에 따라 역할-구성원 바인딩에 변경이 생길 수 있다.

3.2.3 비활성화(deactivation)

커뮤니티 실례가 이미 목적을 달성했지만, 추가로 필요한 경우가 있을 것으로 예상되어 커뮤니티 실례가 사라지지 않고 잠시 비활성화될 수 있다. 실례에 대한 정보는 커뮤니티 관리 프레임웍 내에 저장되고, 커뮤니티 실례가 필요해지면 저장된 정보를 기반으로 활성화에 필요한 작업이 진행된다.

3.2.4 삭제(deletion)

커뮤니티 실례가 더 이상 필요하지 않을 것으로 예상되면 커뮤니티 실례가 삭제된다. 즉, 해당 실례에 대한 정보가 삭제되고, 역할-구성원 바인딩이 모두 해제된다. uT-개체 내에서 동작하던 글루도 삭제될 수 있다. 커뮤니티 실례를 삭제할 것인지 비활성화할 것인지는 커뮤니티 관리 프레임웍에서 결정할 것이다.

3.2.5 발전(evolution)

유비쿼터스 컴퓨팅 기술이 발전하면서, 새로운 종류의 하드웨어, 소프트웨어가 USS 내에 새로 설치될 것이고, 자율적인 커뮤니티의 경우에는 변화하는 환경에 맞추어 커뮤니티 템플릿이 수정되고 가용 uT-개체를 활용하고자 할 것이다. 이러한 경우에, 커뮤니티 실례 내의 기존 역할-구성원 바인딩이 해제되고 새로운 uT-개체를 활용하기 위한 역할-구성원 바인딩이 새롭게 이루어질 수 있다. 즉, 발전된 USS 환경에 맞추어 커뮤니티 실례가 발전된 형태로 재구성된다.

4. 커뮤니티 컴퓨팅의 실현

4.1 사례1: MDA 기반의 커뮤니티 시스템 개발

MDA(model-driven architecture)는 요구 사항 분석에서 획득된 상위의 추상화된 모델로부터 시스템을 개발한다. 그리고, 최종 시스템을 얻기까지 추상화

단계를 낮추어 가면서 개발 과정을 진행해간다. 이러한 기본적인 MDA의 개발 절차를 도입하여 커뮤니티 시스템을 개발한다.

그림 3에서 제시하고 있는 바와 같이, 커뮤니티 컴퓨팅 모델(Community Computing Model, CCM), 개발 환경 독립적인 커뮤니티 구현 모델(Community Computing Model-Platform Independent, CCM-PI), 개발 환경 의존적인 커뮤니티 구현 모델(Community Computing Model - Platform Specific, CCM-PS), 이렇게 세 단계로 커뮤니티 시스템을 구체화한다.

가장 추상화 단계가 높은 CCM에서는 목표로 하는 USS에 대한 커뮤니티 메타포어를 기술한다. '사회(society)'라 불리는 USS에 대한 모델링을 기반으로 USS를 구성하는 커뮤니티를 기술한다. 이를 위해, CML(Community Modeling Language)라는 모델링 언어를 정의하였다.

다음 단계는 CIM-PI이다. 이 단계에서는 시스템의 구성 요소와 구성 요소들 간의 상호 작용에 대한 기술까지 포함해서 보다 자세하게 시스템이 기술된다. 하지만, 아직도 해당 구성 요소들이 개발될 환경에 대한 고려는 반영되지 않는다. CIM-PI의 전체적인 틀은 CCM에서부터 유래되고 구체적인 사항은 개발자에 의해 기술되어야 한다. 이를 위해서, CIL-PI(Platform Independent Community Implementation Language)라는 모델링 언어를 정의하였다.

마지막 단계는, CIM-PS이다. 이 단계에서는 실제 구성 요소들이 개발될 구체적인 환경에 대한 고려가 반영된다. CIM-PS의 기본 틀은 CIM-PI로부터 유래되고, 구체적인 사항들이 개발자에 의해 채워진다. 이를 위해서 CIL-PS (Platform Specific Community Implementation Language)라는 모델링 언어를 정의하였다.

모델 변환 과정이 CCM에서 CIM-PS로 진행되면서 실제 개발 코드의 틀과 개체 간의 협력에 관련된 코드들이 생성된다. 하나의 USS에 대한 모델로부터 개별 uT-개체의 구체적인 코드의 단계로 구체화가 진행되므로 전체적으로 조화되는 uT-개체들을 생성할 수 있다는 장점을 갖는다.

4.2 사례 2: 서비스 종합 기반의 커뮤니티 시스템 개발

커뮤니티를 실제 실행 가능한 서비스들의 동적인 조합으로 간주한다. 동적인 서비스 조합의 연구로는 웹 기반의 서비스 조합에 대한 연구가 이미 활발히 진행되어 왔다[14-16]. 웹 에이전트의 도움으로 기존의 웹 서비스의 의미에 대한 지식을 획득하고 이에 기반해서 동적으로 서비스를 조합한다. 본 연구에서는, 커뮤니티 매니저라는 중앙의 프로세스가 USS 내에서 정의되는 커뮤니티 템플릿에 대한 정보를 갖고 있고, 이 정보를 근거로 필요한 uT-개체를 구성원으로 바인딩하고 필요한 action을 취하도록 조정하는 역할을 담당한다.

커뮤니티 템플릿은 XML 형식의 CDL(Community Description Language)를 사용하여 표현된다. 커뮤니티 매니저는 USS의 컨텍스트와 서비스 사용자의 개인적인 컨텍스트를 파악하고 커뮤니티 템플릿에 주어지는 목적을 해석해서 USS 내의 다양한 서비스 중에 적절한 것을 선택한다. 이러한 기능 수행을 위해 CM은 크게 세 가지 기능 요소를 갖는다. CO(Community

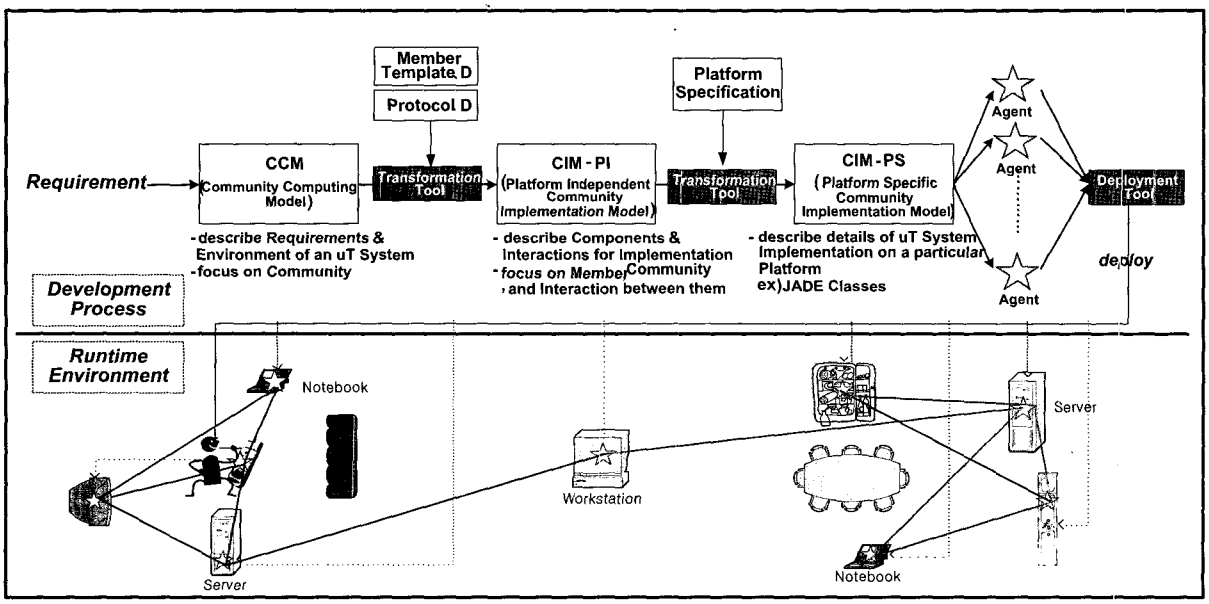


그림 3 MDA 기반의 커뮤니티 컴퓨팅 시스템 개발 과정의 개념도

Organizer)는 커뮤니티 템플릿을 해석하는 기능을 담당하고, CE(Community Executor)는 USS 내의 uT-개체에 대한 역할-구성원 바인딩을 수행하고, SG (Situation Generator)는 USS 내의 컨텍스트 정보를 생성하는 uT-개체와 협력하여 컨텍스트 정보를 수집하고 커뮤니티 실례에 대한 수요를 파악하는 기능을 담당한다.

그림 4는 커뮤니티의 역할-구성원 바인딩이 동적으로 이루어지는 과정을 보여주고 있다. 커뮤니티가 생성되는 단계 (생성 단계), 커뮤니티가 구성되는 단계 (구성 단계), 커뮤니티가 실제 활성화되는 단계 (실행 단계)로 진행해 가면서 해당 USS 내의 구체적인 uT-개체로의 역할-구성원 바인딩이 진행된다.

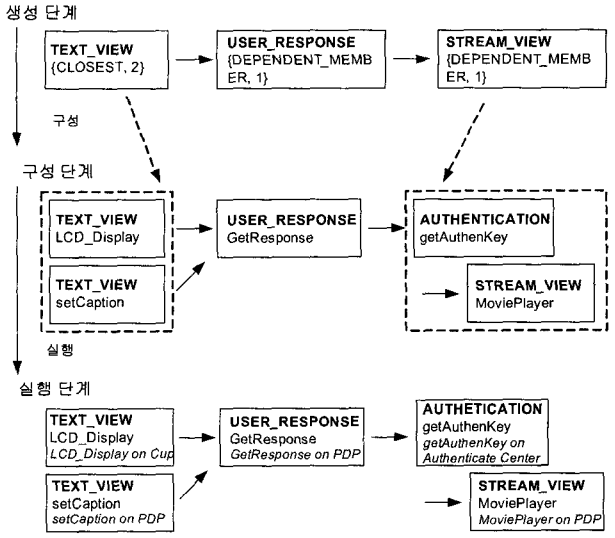


그림 4 커뮤니티 실례의 진행 과정에서의 동적인 역할-구성원-바인딩

5. 결 론

본 연구에서는 협조적인 서비스를 목표로 하고, 또한 이 서비스들이 확장 가능하고 주어진 상황 변화에 적응적으로 대응할 수 있도록 기존의 분산 시스템 모델과는 구별되는 새로운 '커뮤니티 컴퓨팅'을 제시한다.

USS 내의 uT-개체들을 커뮤니티라는 형식으로 조직하고 이들 uT-개체들이 협력하여 사용자를 위한 서비스를 제공한다는 것이 기본적인 가정이다. 커뮤니티의 역동성에 따라 정적인 커뮤니티, 동적인 커뮤니티, 자율적인 커뮤니티로 구분하며, 현재의 연구와 개발은 동적인 커뮤니티 단계에서 진행되고 있으며, 커뮤니티 컴퓨팅에서 궁극적으로 추구하는 바는 상황에 따라 자율적으로 동작하는 자율적인 커뮤니티다.

기존의 분산 시스템에 비해 커뮤니티 컴퓨팅이 갖는 장점은 커뮤니티 템플릿의 확장성과 uT-개체의 재활용성에 있다. 커뮤니티 템플릿은 커뮤니티를 통해 제공하고자 하는 서비스의 목적, 서비스를 수행하기 위한 역할, 역할들의 상호 작용을 정의한 글루를 구성 요소로 갖는다. 글루가 이미 개발된 uT-개체에 설치되어 동작함으로써 uT-개체가 다양한 목적을 갖는 서비스에 참여하게 된다. 즉, 새로운 서비스를 위해 새로운 프로그램을 개발하는 것이 아니라, 이미 설치된 uT-개체들을 활용할 방안을 제시함으로써 새로운 서비스를 제공할 수 있다는 점에서 기존의 분산 모델과 차별화된다고 하겠다.

본 고에서는 커뮤니티 컴퓨팅에 대한 기본적인 개념과 접근 방법을 소개하는 정도이고, 본 연구 과제에서는 커뮤니티 템플릿을 위한 언어에 대한 정의, 커뮤니티 구성원으로서 참여하기 위한 uT-개체의 형상, 커뮤니티 실례에 대한 관리, 개인화된 커뮤니티 템플릿에 대한 정보 보호 등 기술적으로 다양한 분야에서 연구가 진행되고 있다.

참고문헌

[1] D. Garlan, D. Siewiorek, A. Smailagic, and P. Steenkiste, "Project Aura: Toward Distraction-Free Pervasive Computing," IEEE Pervasive Computing, April-June 2002, pp.22-31.

[2] Manuel Román, Christopher K. Hess, Renato Cerqueira, Anand Ranganathan, Roy H. Campbell, and Klara Nahrstedt, "Gaia: A Middleware Infrastructure to Enable Active Spaces," IEEE Pervasive Computing, Oct-Dec 2002, pp. 74-83.

[3] John Barton, Tim Kindberg, "The Cooltown User Experience," HP Technical Report, HPL-2001-22, February 2001.

[4] Bin Wang, John Bodily, and S. Gupta, "Supporting Persistent Social Groups in Ubiquitous Computing Environments Using Context-Aware Ephemeral Group 서비스," Proc. 2nd IEEE International Conference on Pervasive Computing and Communications, March 2004, pp. 287-296.

[5] Mohan Kumar, et al., "PICO: A Middleware Framework for Pervasive Computing," IEEE Pervasive Computing, Vol.2, No.3,

July/September 2003, pp.72-79.
[6] S. S. Yau, F. Karim, Y. Wang, B. Wang, and S.Gupta, "Reconfigurable Context-Sensitive Middleware for Pervasive Computing," IEEE Pervasive Computing, Vol.1 No.3, July-September 2002, pp.33-40.
[7] Microsoft Easyliving, available at http://research.microsoft.com/easyliving/
[8] Philips HomeLab, available at http://www.research.philips.com/technologies/misc/homelab
[9] Information Society Technologies, availale at http://cordis.europa.eu/ist/
[10] IST Advisory Group, Ambient Intelligence: from vision to reality: For participation - in society & business, January 2005. available at http://cordis.europa.eu/documents/documentlibrary/ADS0000806EN.pdf
[11] (재)유비쿼터스컴퓨팅사업단, Ubiquitous Smart Space White Paper, 2005년 11월.
[12] J. Michael Yohe, "Community Computing and the Computing Community," Proc. of ACM SIGUCCS Conference on User Services, October 1994, pp.35-39.
[13] J. Sawamoto, K. Mutoh, H. Tsuji, and H. Koizumi, "Evaluation of Multi-Agent Model for 커뮤니티 Formation in Network Society," Proc. of International Conference on Advanced Information Networking and Application, March 2004.
[14] T. Nishimura, H. Yamaki, T. Komura, and T. Ishida, "Community Viewer: Visualizing Community Formation on Personal Digital Assistants," ACM SIGAPP Applied Computing Review, Vol.6, No.1, February 1998, pp.13-18.
[15] K. Ragab, N. Kaji, and K. Mori, "Service-Oriented Autonomous Decentralized Community Communication Technique for a Complex Adaptive Information System," Proc. of IEEE/WIC International Conference on Web Intelligence, October 2003, pp.323-329.
[16] V. Sunderam, J. Pascoe, and R. Loader, "Towards a Framework for Collaborative Peer Groups," Proc. of IEEE/ACM International Symposium on Cluster Computing and the Grid, May 2003, pp.428-433.
[17] Joaquin Miller and Jishnu Mukerji. Ed. MDA Guide Version 1.0.1, June 2003, available at http://www.omg.org/mda/
[18] Yuna Jung, Jungtae Lee, Minkoo Kim, "Multi-agent based Community Computing System Development with the Model Driven Architecture," AAMAS'06, May, 2006, Hakodate, Japan, p.1329-1331.
[19] Hyeonsook Kim , Yunju Shim, We-duke Cho, "Community Manager, Dynamic Collaboration Solution on Heterogeneous Environment," Proc. of IEEE Conference on Pervasive Services, June 2006, Lyon, France, pp.39-46.
[20] Andrew S. Tanenbaum, et al., Distribued Systems: Principles and Paradigms, Prentice Hall, 2002.

강 경 란

1992 서울대학교 계산통계학과(학사)
1994 한국과학기술원 전산학과(석사)
1999 한국과학기술원 전산학과(박사)
1999~2000 한국정보통신연구원 선임연구원
2000~2002 ㈜디지탈웨이브 책임연구원
2002~2004 한국정보통신대학교 연구교수
2004~현재 아주대학교 정보 및 컴퓨터공학부 조교수
관심분야 : 멀티캐스트, 이동성 관리, 커뮤니티 컴퓨팅
E-mail: korykang@ajou.ac.kr

김 민 구

1977 서울대학교 계산통계학과 전산학(학사)
1979 KAIST 전산학과 전산학(석사)
1989 펜실베니아 주립대학교 전산학(박사)
1981~현재 아주대학교 정보 및 컴퓨터 공학부 교수
1997~1998 한국정보과학회 인공지능 연구회 운영위원장
1999~2000 University of Louisiana, CACS 연구과학자
관심분야 : 인공지능, 데이터 마이닝, 정보검색, 커뮤니티 컴퓨팅
E-mail : minkoo@ajou.ac.kr

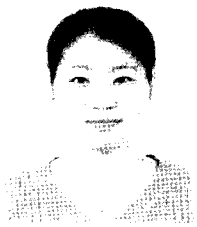
이 정 태

1979 서울대학교 농과대학 원예과(농학사)
1981 서울대학교 자연과학대학 계산학 (이학석사)
1983 서울대학교 자연과학대학 계산학 (이학박사)
1983~1988 울산대학교 전산학과 전임강사
1988~현재 아주대학교 정보통신대학 교수
1988~현재 아주대학교 정보 및 컴퓨터 공학부 정교수

관심분야 : 컴포넌트 베이스 시스템, 미들웨어, 객체지향 응용 프레임웍

E-mail : jungtae@ajou.ac.kr

정 유 나



2000 아주대학교 정보및컴퓨터공학부 (공학사)
2002 아주대학교 정보통신전문대학원 (공학석사)
2005 아주대학교 정보통신전문대학원 공학박사 수료
2005~현재 유비쿼터스 프로젝트(커뮤니티 컴퓨팅 모델)

관심분야 : 멀티 에이전트 시스템, 커뮤니티 컴퓨팅

E-mail : serazade@ajou.ac.kr

조 위 덕

1983~1987 한국과학기술원 전기 및 전자공학과(박사)
1983~1990 금성전기(현LG전자) 기술연구소 DSP 연구실장
1990~1991 한국생산기술연구원 전자정보시스템연구부 팀장/조교수
1991~2003 전자부품연구원 시스템연구본부 본부장
2003~현재 유비쿼터스컴퓨팅사업단 단장, 아주대학교 전자공학부 교수

관심분야 : 유비쿼터스 컴퓨팅/네트워크, 센서 네트워크, Post-PC (차세대 Smart PDA), Interactive DTV 방송기술, 고품질 홈서버/게이트웨이기술, 디지털방송/이동통신 연계 융합플랫폼기술, 무선인터넷응용기술

E-mail : chowd@ajou.ac.kr

김 현 숙

1996~2001 한동대학교 전산전자공학부 (학사)
2001~2003 키스톤테크놀러지 SI팀 연구개발
2003~2004 비버텍 코리아 ERP 시스템 컨설팅
2004~현재 유비쿼터스 시스템 연구센터, 유비쿼터스 지능공간 솔루션팀 선임연구원

관심분야 : 유비쿼터스 컴퓨팅, 커뮤니티 컴퓨팅, 상황인지, 협업 솔루션, 시맨틱 웹서비스

E-mail : virtus78@ajou.ac.kr