의미 단일화를 통한 충돌 해결 방법[1)]

이건수° 이건선 김민구
아주대학교 정보통신 전문 대학원°
{lks7256°, kslee, minkoo}@ajou.ac.kr

A Conflict Prevention Method using Semantic Unification

Keonsoo Lee°, Keonsun Lee, Minkoo Kim
Graduate School of Information and Communication, Ajou University°

요 약

유비쿼터스 컴퓨팅 환경에서 서비스는 개별 사용자의 요구에 의거해 제공되어야 한다. 동일 환경에서 다수의 사용자가 원하는 서비스들이 서로 상충되는 관계를 갖고 있을 때, 충돌이 발생한다. 이러한 충돌이 발생하는 근본 적인 원인은, 서비스 선택 과정에서 찾을 수 있다. 일반적인 서비스 선택과정은 모델링된 환경 안에서 각 사용자들의 개별 프로필을 통해, 그 사용자에게 적합한 서비스를 매칭하는 흐름을 따른다. 문제는 동일한 환경 정보가 각각의 사용자에게 다른 의미로 적용될 수 있다는 점이다. 즉, 현재 온도가 18°C 라고 할 때, 춥게 느끼는 사용자가 있을 수 있고, 반대로 덥게 느끼는 사용자가 있을 수도 있다. 이 경우 각각의 사용자에 대해 온도를 높여주는 서비스, 온도를 낮춰주는 서비스가 동시에 수행되게 되고, 충돌은 발생하게 된다. 이에 본 논문에서는 환경을 모델링함에 있어서, 센서 정보에 대한 의미를 단일화 함으로써, 동일한 환경 정보가 상반되는 서비스의 조건으로 동작하는 것을 방지하는 방법을 제안한다. 이 방법은 기존의 시스템 오브젝트 혹은 에이전트들 간의 협상 방법보다 통신 작업량을 줄여주고, 환경 모델의 직관적인 구조를 제공함으로써, 보다 효율적인 충돌 방지를 가능하게 해 준다.

1. 서 론

의견이나 이해가 맞지 않아 서로 맞서는 경우, 충돌은 발생한다. 사람들 사이의 충돌은 항상 발생하기 마련이다. 인류 역사는 이러한 충돌을 해결하기 위한 방법의 역사로 볼 수 있다. 헤겔의 변증법 역시 정(正)과 그것의 반(反)의 충돌을 해결하여 합(合)에 이르는 충돌 해결의 방법으로 볼 수 있다. 이 같은 상충되는 이해 사이의 충돌과 그것의 해결 방법은 오늘날 컴퓨팅 시스템에서도 발생하고 있다. 특히 유비쿼터스 컴퓨팅 환경에서는 개별 사용자에게 특성화된 서비스를 제공하는 과정에서 개별 사용자의 욕구에 따른 충돌을 처리하는 것이, 시스템에 대한 사용자 만족도를 보장할 수 있는 기본 조건이 된다. 이는 사용자가 자동화된 서비스를 제공 받는 입장에서, 잘못된 서비스를 제공받는 것은 그 시스템에 대한 신뢰도에 치명적인 영향을 미치기 때문이다.

유비쿼터스 컴퓨팅 환경에서 충돌이 발생하는 원인은, 그 서비스가 제공되는 과정에서 찾아볼 수 있다. 일반적으로 서비스가 제공되기까지의 흐름은 다음의 단계를 따르고 있다. 우선, 시스템은 환경에 분포된 센서들을 통해 현재 환경에 대한 내부 모델을 작성하게 된다. 환경에 대한 모델링 작업이 완료되면, 그 환경에 존재하는 사용자의 특성을 파악하게 된다. 사용자의 특성을 인지하는 과정은 그 사용자의 프로필을 어디에서 관리하고 있는가에 영향을 받기도 하지만, 보다 중요한 것은, 현재 시스템이 제공할 수 있는 서비스가 무엇인가에 더 큰 영향을 받는다. 가령, 조명 서비스만이 가능한 환경이라면, 사용자의 위치 정보만으로도 충분할 것이고, 채널을 조절하는 서비스를 제공한다면, 그 사용자가 선호하는 프로그램 정보까지도 필요할 것이다. 이러한 사용자의 프로필에 근거하여, 현재 상황에서 제공 가능한 서비스들을 찾아내고, 가장 적합한 서비스를 선택해 수행하게 된다. 이때, 어떻게 서비스를 선택하는 가에 대한 문제는 본 논문의 범위를 벗어나기 때문에 고려하지는 않지만, 보통, 최종 선택의 경우에는 사용자의 명시적인 선언을 받거나, 서비스별 중요도를 계산하는 방법이 주로 사용된다.

이렇게 선택된 서비스를 수행하는 과정에서 충돌은 발생하게 된다. 충돌은 서로 상반되는 서비스를 동시에 수행하려고 하기 때문에 발생하지만, 보다 근본적인 원인은, 동일한 환경에서 상충되는 관계의 서비스가 동시에 발생하게 된다는 점이다. 이는 동일한 환경 모델을 각각의 사용자에 프로필에 적용하는 과정에서 동일한 환경이 각기 다른 의미로 인지되기 때문이다.

본 논문은 환경 요소의 의미를 단일화시킴으로써, 개별 사용자 프로필에 근거한 환경 모델 인지의 차이를 제거하고, 이를 통해 상충되는 서비스의 동시 발현을 방지하는 방법을 제안한다.

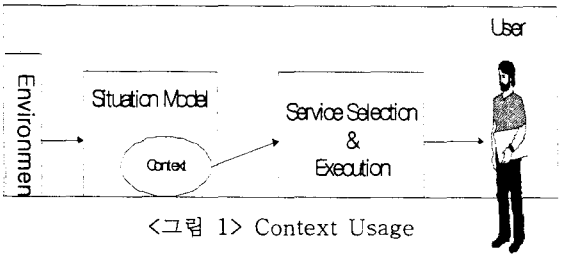
2. 관련연구

2.1 Context Acquisition

사용자로부터 명시적인 명령을 받지 않고, 스스로 적절한 서비스를 제공하는 유비쿼터스 컴퓨팅 시스템에서는

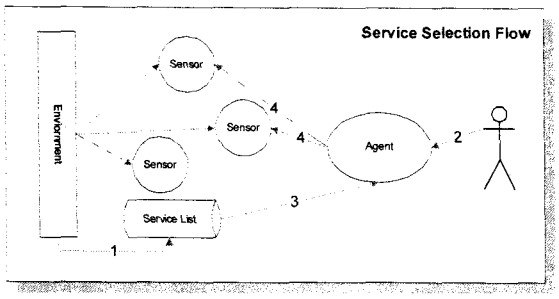
1) This research is supported by the ubiquitous Autonomic Computing and Network Project, the Ministry of Information and Communication (MIC) 21st Century Frontier R&D Program in Korea.

현재 상황을 인지하는 것이 무엇보다 중요하다. 서비스를 제공하기 위한 실행 상황은 흔히 컨텍스트라고 일컬어지는데, 이에 대한 명확한 정의는 존재하지 않지만, 일반적으로 개체들의 상황을 특징지을 수 있는 모든 정보를 의미한다 [1]. 이러한 컨텍스트를 획득하기 위해서는 다음의 3가지 방법을 사용할 수 있는데, 첫째, 센서를 통한 정보 획득, 둘째, 논리적인 추론의 결과로서 정보 획득, 그리고 마지막으로 사용자로부터의 명시적인 정보 제공이 그것이다. 이렇게 획득된 컨텍스트를 통해 현재 상황에서 제공되어야 하는 서비스가 결정되고, 그 서비스의 속성이 조절된다. 즉, 이 때, 개별 서비스가 수행되기 위해 필요로 하는 컨텍스트는 전체 환경 정보중에서 그 서비스의 시작 혹은 종결 조건을 검사하기 위해 필요로 하는 하위 집합으로 정의 될 수 있다. 가령, 사용자가 원하는 TV 채널을 보여주기 위해서, 현재 습도나 밝기에 대한 환경 정보는 필요하지 않다.



<그림 1> Context Usage

그렇기 때문에, 컨텍스트의 범위를 정의하는 주체는 현재 발현되려는 서비스가 되어야 한다. 이는 현재 상황이 특정 서비스를 발현 시키는 것이 아니라, 특정 서비스가 동작하기 위해서 현재 상황을 검색하고, 그 상황이 자신의 수행 조건에 일치한다면 구동하는 흐름을 따라야 함을 의미한다 [3]. 그러나 이를 위해서는 개별 서비스를 제공하는 에이전트가 따로 존재하여야 하고, 그 에이전트는 계속해서 상황 정보의 변화를 감시해야만 한다. 효율적인 서비스 선택, 제공 및 환경 정보 검색을 위해서는 <그림 2>에서와 같은 흐름을 따라야 한다. 우선 주어진 환경에서 제공 가능한 서비스 목록을 작성하고, 그 환경에 특정 사용자가 등장하게 되면, 에이전트는 서비스 리스트와 그 사용자의 프로파일에 근거하여 제공 가능한 서비스들을 추출한다. 그리고, 추출된 각각의 서비스들에 대해, 그 서비스가 필요로 하는 환경 정보를 확인하여, 그 서비스의 수행 여부를 결정한다 [7].



<그림 2 > Service Selection Flow

물론, 이 경우 센서 정보 없이 사용자의 프로파일 정보만으로 제공 가능한 서비스 목록을 추출해야 한다는 단점이 있지만, 각각의 에이전트가 전체 환경에 대한 상황 모델을 관리하지 않아도 된다는 잇점이 있다. 또한, 유비쿼터스 컴퓨팅 환경에서 제공되는 서비스의 신뢰성을 고려할 때, 원하지 않는 서비스의 발생은 원하는 서비스가 자동으로 제공되지 않는 것 보다 사용자의 불만을 유발시킬 가능성이 크다. 이는 곧, 주어진 상황에서 발생 가능한 서비스들 중 사용자에게 적합한 서비스를 제공하는 것보다 사용자에게 적합한 서비스들 중, 그 상황에서 제공 가능한 서비스를 찾는 것이 시스템의 신뢰도를 높이는 방법임을 말해준다.

2.2 Conflict Resolution

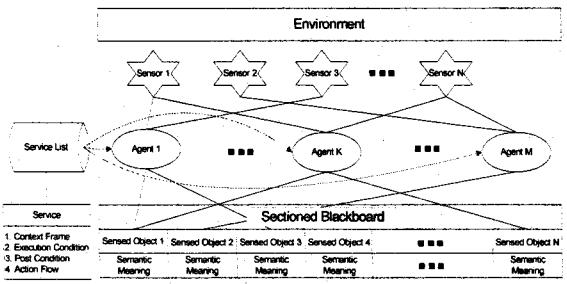
충돌을 해결하기 위한 방법은 크게 다음의 두 가지 방법으로 나눌 수 있다. 첫 번째 방법은, 충돌이 발생하기 이전에 충돌 가능성이 있는 서비스를 사전에 방지하는 방법으로, 하나의 서비스가 선택되었을 때, 그에 상반되는 다른 서비스들의 발현을 방지하는 방법이다. 이를 위해서, 시스템 내의 서비스 수행 주체들은 다른 서비스들의 실행 상황을 알고 있어야 하고, 그 서비스가 발현되기까지의 상황을 인지하고 있어야 한다. 두 번째 방법은 서비스가 실행되는 순간에 상충되는 서비스들 사이의 교섭과정을 거쳐, 서비스들 간의 충돌 없는 수행을 결정짓는 방법으로 서비스 간 우선순위를 바탕으로 대기, 대체, 중지 등을 결정하는 방법이다[4].

이처럼 충돌 해결 방법이 나뉘는 원인은 충돌의 원인에 근거한다. 첫 번째 방법이 해결할 수 있는 충돌은, 서비스를 선택해서 수행할 때, 그 서비스의 수행으로 인해 변화되는 환경의 상태가 다른 서비스의 수행 조건 혹은 수행 목표에 영향을 끼치는 충돌이다. 이 경우 서비스는 수행되기 위해 필요한 장치의 사용, 서비스 수행상의 순차적 흐름 등은 고려하지 않는다. 반면 두 번째 방법으로 해결되는 충돌은 첫 번째 방법이 고려하지 않는 문제들을 처리한다. 서비스를 수행하기 위해 필요로 하는 장치가 이미 다른 서비스에 의해 운용되고 있는 경우, 동일한 자원을 여러 서비스가 동시에 사용하고자 하는 경우, 하나의 서비스를 수행하기 위해 필요한 자원을 어떤 순서로 사용할 것인지를 결정하는 과정에서 발생하는 충돌 해결방법이다. 유비쿼터스 컴퓨팅 환경에서 발생하는 충돌은 그 원인에 의해 구분지어질 수 있고, 이 충돌 원인에 의해 그 해결 방법 또한 달라진다.

3. 충돌 해결 방법

본 논문에서 제안하는 충돌 해결 방법은 위에서 언급한 첫 번째 충돌 타입의 해결 방법으로, 서비스가 발현되는 과정에 있어 다른 서비스와의 관계를 기반으로 기존에 수행되는 서비스와의 충돌을 방지하는 방법이다. 이를 위해 본 연구에서 제안하는 방법은 각 서비스가 필요로 하는 컨텍스트 정보의 의미를 통일시킨다. 즉 시스템에서 서비스를 선택하기 위해, 그 컨텍스트를 인지하는 과정에서, 받아들이는 상황 정보의 의미를 단일화 함으로써, 또 다른 서비스가 인지하는 상황 정보를 다른 의미

로 받아들이는 것을 방지한다. 동일한 상황 정보를 필요로 하는 서비스들 사이의 의미 단일화는 상충되는 서비스의 발현을 서비스 선택의 단계에서 방지하기 때문에, 추가적인 교섭 작업이 필요하지 않고, 서비스 오브젝트들(에이전트)들 사이의 의미 교환을 위한 통신 작업을 줄일 수 있다. <그림 3>은 본 논문에서 제안하는 유비쿼터스 컴퓨팅 시스템과 의미 단일화를 위한 블랙보드의 활용 모델을 보여준다. 모델은 각 사용자별 에이전트들과, 모델의 적용 환경에 영향을 받는 서비스 리스트, 환경 정보를 인식하는 센서들 그리고, 의미 단일화를 위한 블랙보드로 구성된다. 우선, 서비스 리스트에는 주어진 환경에서 제공 가능한 서비스들의 목록을 갖고 있다. 각각의 서비스는 동작 가능한 상황에 대한 실행/중지 조건, 서비스를 수행하기 위한 플랜, 그리고 상황 프레임 정보로 구성되어 있다. 이는 곧, 특정 서비스가 수행되기 위해서는 어떤 조건(context frame)하에서 어떤 장치(which devices)들을 어떤 순서로 (action flow) 사용하고, 언제 서비스를 종료해야 하는지 (Post condition)를 정의하고, 그 서비스가 수행되는 동안 환경에 어떤 영향을 미치고 있어야 하는지(execution condition)에 대한 정보로 이루어져 있다. 각 에이전트는 자신이 담당하고 있는 사용자의 요구를 직/간접적으로 인지하고, 현재 환경에 근거해 수행되어야 하는 서비스를 선택한다. 이 때, 충돌이 발생할 수 있는데, 충돌 발생의 원인은 센서를 통해 받아들인 환경 데이터 값을 다른 의미로 인지하기 때문이다.



<그림 3> Proposed Model

가령, 센서를 통해 감지된 현재 온도가 21도라고 할 때, 사용자 A를 맡고 있는 1번 에이전트는 현재 환경이 춥다고 판단해 히터를 가동시키고, 사용자 B를 담당하는 2번 에이전트는 21도를 더운 환경이라고 판단해 에어컨을 가동할 수 있다는 의미다. 즉 동일한 상황 정보를 이용하는 상이한 서비스가 센서를 통해 들어온 원자료(raw data)를 각기 다른 의미로 이해했을 경우, 상반되는 서비스들 사이에 충돌이 발생하게 된다. 이를 방지하기 위해, 본 모델에서는 블랙보드 메모리를 사용한다. 각 에이전트는 센서를 통해 들어온 데이터의 의미를 다른 에이전트들과 공유하기 위해, 자신이 이해한 의미를 블랙보드에 공지하게 된다. 가령, 현재 온도 21도에 의해 에이전트 A가 에어컨을 작동시켰다면, 이 에이전트는 에어컨 서비스를 수행하는 동시에, 현재 온도의 의미로 "더움"이라는 내용을 블랙보드에 작성하게 되고, 다른 에이전트가 온도 관련 서비스를 수행하려고 할 때, 이 블랙보드를 참조함으로써, 21도는 "더움"이라는 의미로 다른 에이전트에 의해 사용되고 있으므로, "춥다"라는 의미로 해석되어 에어컨 등의 서비스가 동작하는 것을 방지한다. 이 블랙보드 메모리는, 센서가 인지하는 데이터의 타입에 따른 분리된 구획을 갖고 있기 때문에, 실제 에이전트는 자신이 참조하려는 센서 정보의 의미를 보다 효율적으로 탐색할 수 있다.

이 흐름을 제안된 모델에 적용하면, 다음의 결과를 얻을 수 있다. 우선 각각의 에이전트는 자신이 담당하고 있는 사용자의 프로파일 정보에 근거하여, 서비스 리스트로부터 제공 가능한 서비스 목록을 찾는다. 사용자에게 제공될 수 있는 서비스가 온도조절 서비스라고 할 때, 에이전트는 서비스 리스트에서 가져온 온도 조절 서비스의 정보에 근거하여, 그 서비스가 필요로 하는 컨텍스트 정보를 알 수 있다. 이 정보에 의해 환경에 분포도니 센서들로부터 현재 상황 정보를 찾아온다. 온도조절 서비스의 경우라면, 현재 온도, 시간, 사용자의 위치 등이 그 컨텍스트 정보가 된다. 센서로부터 받아온 정보가 "사용자가 낮 2시에 거실에 있고, 현재 21도다" 라고 할 때, 이 상황이 에어컨을 사용해야 할지, 히터를 사용해야 할지를 말해주지는 않는다. 필요한 것은 이 온도가 지금 추운 온도인지, 더운 온도인지에 대한 의미이다. 에이전트는 블랙보드 메모리를 검색해 현재 온도의 의미가 무엇인지를 검색한다. 이 전에 다른 사용자에 의해 에어컨이 동작했더라면, 이 블랙보드에는 온도 정보에 대해 "덥다"라는 의미를 남겨두었을 것이고, 히터를 사용했더라면, "춥다"라는 의미를 남겨두었을 것이다. 만약, 이전에 온도 조절 서비스가 한번도 수행되지 않았다면, 의미는 정해지지 않았을 것이고, 이때, 서비스는 사용자의 프로파일 정보에 근거하여 자신이 의미를 정의하고, 이렇게 정의된 의미는 다른 사용자의 서비스 수행에 참조 정보로 사용되게 된다.

물론, 블랙보드 메모리를 사용해서 감지된 원자료의 의미를 조율하는 작업이 필요하다. 위의 예에서 21도의 온도를 춥다고 봐야 하는지, 혹은 덥다고 봐야 하는지 어떻게 결정할 것인가의 문제는 본 논문의 범위를 벗어나기 때문에 원자료의 의미를 결정짓는 방법에 대해해서는 고려하지 않도록 한다. 다만, 일반적인 교섭 방법으로 사용되는 우선순위를 사용하는 방법을 택할 수도 있고, 시간 논리를 이용하여 자동으로 의미를 정의할 수도 있다. 이런 방법은 실제적으로 제공되는 서비스들의 종류와 시스템이 동작하는 환경, 그리고 그 환경안에서 서비스를 제공받는 사용자들의 특성에 많은 영향을 받을 수 밖에 없기 때문에 시스템 디자이너와 실 사용자의 요구가 적용될 수 있도록 유연하게 변경 가능할 수 있어야 한다.

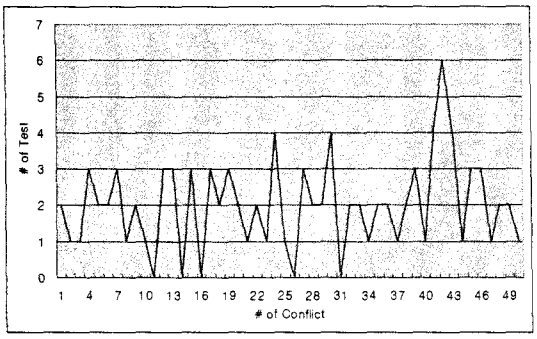
4. 시뮬레이션 및 결과

본 논문에서 제안하는 방법의 효율성을 검증하기 위한 시뮬레이션으로 가상의 시나리오를 가정하고, 그 시나리오 안에서 발생하는 충돌 회피 과정을 보여주고, 이 과정의 복잡도를 계산하였다.

<표 1> 시뮬레이션을 위한 가정

Feature	Value
Sensor Type	3
Service Type	5
# of sensing data for each service's context	2
# of features that the service affects	1

시뮬레이션을 위한 시나리오는 다음과 같다. 우선 시나리오를 위한 가정은 환경 정보를 인지하기 위한 3가지 종류의 센서가 있고, 5개의 서비스가 제공 가능하다. 각각의 서비스는 2가지 종류의 센싱 정보를 컨텍스트로 사용하고, 각각의 서비스가 수행될 때는 센서들이 인지하는 정보중 하나에 영향을 미친다고 가정한다. 서비스들 사이에 충돌이 발생하는 경우는, 동일한 컨텍스트를 사용하는 서비스들 사이, 다른 서비스가 영향을 미치는 환경 정보를 컨텍스트로 사용하는 서비스 사이에서 발생한다고 가정한다. 이때, 각각의 서비스는 1~5분 사이의 지속 시간을 갖고, 동일한 서비스는 0~4분 사이의 간격으로 발생한다. 이러한 가정을 바탕으로 5개의 서비스들을 30분 동안 무작위로 발생시켰을 때, 충돌 조건에 일치하는 서비스 발생은 <그림 4>에서와 같다.



<그림 4 > 시나리오상에서의 충돌 예상 수

이때, 본 논문에서 제안한 방법을 사용하면, 각 서비스의 실행 정보 2개와 서비스 결과 정보 하나를 검색하는 것만으로서 충돌 가능성의 서비스의 발생을 방지할 수 있다. 즉 N개의 서비스가 M개의 환경 정보의 종류가 존재하는 공간에서 K개의 정보로 구성된 컨텍스트를 사용한다고 할 때, 통일된 의미 정보를 사용한다면, 서비스들 사이의 배타관계를 모르더라도, O(N*K)의 탐색만으로 충돌 예측 및 회피가 가능하다.

5. 결 론

본 논문에서는 유비쿼터스 컴퓨팅 환경에서 상황 정보의 의미 단일화를 통해 발생 가능한 서비스 충돌 해결 방법을 제안하였다. 서비스 제공의 근거가 되는 상황 정보를 추상화시켜, 또 다른 서비스가 같은 상황 정보를 사용하는 경우, 인지된 원자료의 의미를 공유함으로써, 같은 정보가 다른 의미로 사용되는 것을 방지함으로 상이한 서비스가 동일한 상황에서 동시에 발현되는 충돌을 예방한다. 실제 환경을 모사하는 시스템의 내부 모델을 구축하는 과정에서 센서를 통해 들어오는 정보의 의미를 통일시킴으로써, 개별 서비스들의 수행 및 종료 조건의 혼동을 방지하기 때문에, 보다 추상적인 단계에서 시스템을 운용할 수 있고, 기존의 서비스 생성 후 조율하는 방법보다 계산 복잡도 및 통신 작업량 줄일 수 있다.

본 방법이 실제 어플리케이션으로 적용되기 위해서는 의미를 정의하기 위한 어휘 집합에 대한 정의 및 의미 결정 과정에 대한 연구가 추가로 수행되어야 한다. 가령 온도 정보의 의미를 어떤 단계로 나눌 것인지, 현재 상황의 의미를 어떻게 정의할 것인지에 대한 결정이 선행되어야 한다. 그러나 이런 문제는 전적으로 그 환경에서 어떤 서비스가 제공 가능하고, 어떤 환경 정보를 인지할 수 있고, 그 서비스를 제공받는 사용자가 어떤 특성을 갖고 있는지 즉 본 모델을 적용하기 위한 도메인에 영향을 받게 된다.

이처럼, 적용되는 도메인과 그 도메인을 얼마나 체계적으로 조직하는가에 따라 시스템의 효율이 영향을 받지만, 환경으로부터 인지된 정보에 의미를 부여하고, 또 이 의미를 공유함으로써, 주어진 도메인에서 보다 효율적인 충돌 해결을 기대할 수 있다.

참고 자료

[1] A. K. Dey, "Providing Architectural Support for Building Context-Aware Applications". PhD thesis, College of Computing, Georgia Institute of Technology, 2000

[2] Reiter, R. "The frame problem in the situation calculus: a simple solution (sometimes) and a completeness result for goal regression." Artificial intelligence and mathematical theory of computation: papers in honor of John McCarthy 359--380.

[3] K Lee and M Kim "Conflict Resolution Method for Multi-Context Situation" PRIMA 2005 Malaysia pp.285-294

[4] K Lee, W Kim and M Kim "Resource Allocation in Multi-Agent System for Ubiquitous Computing Service" PRIMA 2005 Malaysia pp.113-120

[5] H. Levesque, F. Pirri, and R. Reiter "Foundation for the situation calculus" Electronic Transactions on Artificial Intelligence, 2(34)159-178

[6] A K Dey, G D Abowd and D A Salber "A Context-based Infrastructure for Smart Environ-ment", MANSE99 pp.112-128

[7] K. Lee and M. Kim "Service Selection Model using Situation in Ubiquitous Computing Environment" HCI2006