상황인지 응용서비스를 위한 다중사용자간 서비스 충돌 해결

신춘성, 우운택

광주과학기술원 U-VR 연구실

Conflict Resolution among Users for Context-Aware Applications

Choonsung Shin, Woontack Woo

GIST U-VR Lab.

{cshin, woo}@gist.ac.kr

요 약

본 논문은 상황인지 응용 서비스가 제공되는 환경에서 다중 사용자 간의 서비스 충돌 해결을 위한 컨텍스트 관리기를 제안한다. 사용자 간의 서비스 충돌은 서비스 영역 내의 상황인지 서비스를 다수의 사용자들이 동시에 사용함으로 인해 서비스가 적절하게 제공되지 못하기 때문에 발생한다. 제안된 컨텍스트 관리기는 사용자, 시간, 장소, 제스처, 의도 등의 컨텍스트를 활용하여 사용자 간의 우선순위를 결정하기 때문에 사용자의 상황에 따라 사용자를 동적으로 선택한다. 또한, 컨텍스트 관리기는 히스토리 활용하여 사용자들의 서비스 충돌해결 형태를 습득하므로 사용자들의 서비스 사용 성향에 따라 우선순위를 조절한다. 그리고 응용 서비스의 특징 정보를 활용하므로 응용서비스에 따라 우선 순위를 다르게 적용한다. 마지막으로, 제안된 사용자 간의 서비스 충돌 해결 기법을 스마트 홈 테이스베드인 유비홈에서 적용하여 유용함을 보인다. 따라서 제안된 사용자 간의 충돌해결 기법은 상황인지 응용 서비스들이 다수의 사용자를 대상으로 서비스를 제공하는데 중요한 역할을 할 것으로 기대된다.

1. 서 론

상황인지 응용 서비스는 일상 생활속의 사물에 편재되어 있는 다양한 컴퓨팅 자원으로부터 얻어진 환경과 거주자의 정보를 토대로 상황을 인지하여 사용자에게 적절한 서비스를 제공한다 [1]. 사용자들은 이러한 환경에서 자신에게 맞추어진 서비스를 편리하게 제공받는다. 그러나 다수의 사용자들이 이러한 상황인지 응용 서비스들을 공유하게 되면 이들 간에는 서비스 충돌이 발생한다. 사용자 간의 서비스 충돌은 상황인지 응용 서비스를 여러 사용자들이 동시에 접근함에 따라 이들에게 각각 서비스를 제공해야 하는 상황이다. 이로 인해 응용 서비스는 서비스 실행 결정을 적절하게 내릴 수 없게 되며, 사용자들은 자신에 맞는 상황인지 서비스를 제공 받기 힘들다. 따라서, 상황인지 응용 서비스들이 다수의 사용자들을 대상으로 서비스를 적절하게 제공하기 위해서는 사용자 간의 서비스 충돌을 반드시 해결해야 한다.

상황인지 응용서비스를 위한 연구는 스마트 홈과 지능형 사무실을 대상으로 활발히 진행되고 있다. ReBa 는 활동이론을 바탕으로 하고 있으며 지능형 공간 내의 활동, 예를 들면 회의, 발표, 영화 시청, 을 중심으로 서비스들이 적절하게 반응하도록 하고 있다 [3]. Context Toolkit 은 응용서비스가 Aggregator 나 Widget 에 등록될 때 설정한 조건이 일치하면 callback function 을 통해 컨텍스트를 전달한다 [2].

그러나 상황인지 응용서비스가 다수의 사용자에게 적절하게 제공되기 위해서는 다음의 사항을 고려하여 컨텍스트를 관리해야 한다. 먼저, RCSM 과 ReBa 처럼 상황인지 프레임웍 내에서 응용 서비스의 실행 결정을 내리는 경우 다수 사용자의 서비스 요구를 처리해야 한다 [3]. 그리고 Context Toolkit 처럼 응용 서비스가 실행 결정을 내리는 경우 응용서비스에서 다수 사용자에 대한 서비스 정책을 가지거나 최종 컨텍스트를 전달하기 전에 다수 사용자에 대한 충돌을 해결해야 한다 [2].

본 논문에서는 다수 사용자 간의 서비스 충돌을 해결하기 위한 컨텍스트 관리기를 제안한다. 제안된 컨텍스트 관리기는 컨텍스트 변환과 필터링을 수행하는 컨텍스트 전처리모듈, 사용자 간의 서비스 충돌을 해결하는 사용자 충돌 관리모듈, 그리고 응용서비스에 전달될 컨텍스트를 생성하는 최종 컨텍스트 생성모듈로 구성된다

따라서, 제안된 컨텍스트 관리기는 다음과 같은 장점을 갖는다. 첫째, 사용자 간의 우선 순위를 사

* 본 연구는 삼성전자의 Digital Solution Center 의 지원으로 수행되었음.

© JCCI 2005

용자, 시간, 장소, 제스쳐, 그리고 의도 등의 사용자의 컨텍스트를 활용하여 부여하므로 상황에 따라 다르게 사용자 간의 충돌을 해결한다. 또한, 사용자들의 충돌 히스토리를 활용하여 컨텍스트의 각 특징의 가중치를 설정하므로 서비스 충돌해결 성향에 따라 우선 순위가 조절된다. 또한, 서비스의 특징 정보를 활용하므로 응용서비스에 따라 사용자에 대한 우선순위가 다르게 적용된다.

본 논문은 다음과 같이 구성된다. 2 장에서는 정형화된 컨텍스트 기반 서비스 모형에 대해서 기술하고 3 장에서는 사용자 간의 서비스 충돌 해결을 위한 컨텍스트 관리기에 대해서 기술한다. 4 장에서는 실험 및 결과에 대해서 논의한다. 그리고 5 장에서 결론을 맺는다.

2. 정형화된 컨텍스트기반 응용서비스모델

개인화된 서비스를 제공하기 위해 사용자와 주변환경에 대한 상황정보의 일부분을 나타내면서 사용자에 대한 정보를 포괄적으로 표현하기 위한 방법으로써 사용자의 상황 정보를 5W1H(Who, What, Where, When, How, Why) 로 나타내며, 이것을 정형화된 컨텍스트로 정의한다 [4]. 또한 각 요소는 세부 항목을 가짐으로써 사용자에 대한 컨텍스트를 보다 자세하게 표현할 수 있다. 5W1H 로 표현되는 정형화된 컨텍스트는 센서와 서비스 사이의 독립성을 보장하고 생성된 컨텍스트가 여러 서비스에 의해 재 사용되는 장점을 갖는다. 특히 정형화된 컨텍스트 형식은 동일한 컨텍스트를 서비스에 따라 다른 형태로 변경하는 등의 추가 관리 기능을 줄임으로서 센서와 여러 서비스들 사이에 동시 연결성을 보장한다.

ubi-UCAM (Unified Context-aware Application Model for ubiquitous computing environment)은 유비쿼터스 컴퓨팅 환경에서 독립된 여러 종류의 센서와 서비스들이 컨텍스트를 이용한 상호작용을 통해 개인화된 서비스를 제공하기 위한 컨텍스트 기반 응용 서비스 모형이다 [6]. ubi-UCAM 은 그림 1 과 같이 유비센서와 유비서비스로 구성된다.

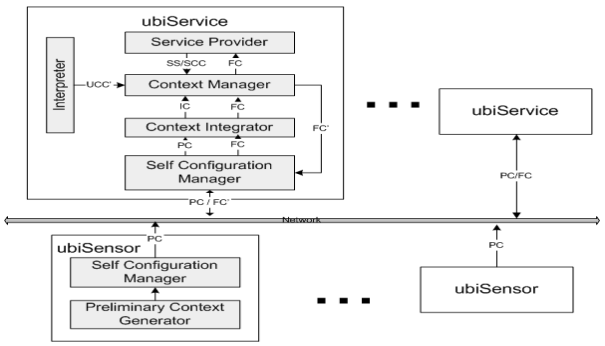

그림 1. ubi-UCAM

유비 센서는 사용자 및 사용자 환경에 대한 변화를 감지하는 센서 모듈과 감지된 정보를 초벌 컨텍스트로 생성하는 초벌 컨텍스트 결정모듈로 구성된다. 유비센서는 센서 모듈에 따라 5W1H 의 일부분 또는 전체를 초벌 컨텍스트 형식으로 생성하여 유비서비스들로 전달한다. 유비서비스는 컨텍스트 통합기, 컨텍스트 관리기, 해석기, 그리고 서비스 관리기로 구성된다. 컨텍스트 통합기는 유비서비스의 동일한 동작 영역에 있는 유비센서들로부터 초벌 컨텍스트를 일정 시간 간격으로 수집하고, 5W1H 를 항목별로 분류한다. 그리고 각 항목 별 특성에 맞는 의사 결정 기법을 적용하여 통합 컨텍스트를 생성한다. 컨텍스트 관리기는 특정서비스 모듈의 동작 조건을 관리하는 해쉬테이블에서 컨텍스트 통합기에서 생성한 통합 컨텍스트와 일치되는 컨텍스트 조건을 검색하고 그에 맞는 서비스를 실행할 수 있는 컨텍스트를 생성한다. 서비스 관리기는 유비서비스가 제공하는 서비스 모듈이 구현된 코드형태로 관리하고 컨텍스트 관리기로부터 실행서비스 이름과 서비스 실행에 필요한 정보인 최종 컨텍스트를 전달받아 서비스를 직접 실행시킨다.

3. 컨텍스트 관리기

컨텍스트 관리기는 서비스 환경, 응용 서비스, 그리고 사용자들로부터 컨텍스트를 수집하고 이를 토대로 응용서비스에서 이용될 컨텍스트를 생성하고 이를 전달한다. 또한 다수의 사용자 간에 발생되는 서비스 충돌을 해결하여 응용서비스가 다수 사용자 환경에서도 서비스를 제공하도록 한다. 이를 위해 컨텍스트 관리기는 컨텍스트 전 처리기, 컨텍스트 데이터베이스, 서비스 충돌관리기, 그리고 최종 컨텍스트 생성기로 구성된다.

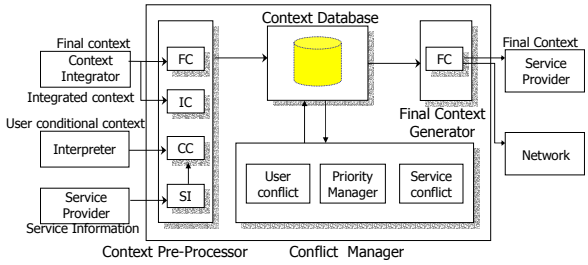

그림 2. 컨텍스트 관리기

그림 2 와 같이, 컨텍스트 전처리기는 다른 외부로부터 입력되는 다양한 컨텍스트에 대한 전처리를 수행한다. 그리고 그 결과는 컨텍스트 데이터 베이스에 반영된다. 충돌 관리기는 전처리된 컨텍스트를 기반으로 사용자 간의 충돌을 해결한다. 그리고 처리한 결과를 컨텍스트 데이터 베이스에 반영한다. 마지막으로 최종 컨텍스트 생성기는 최종 컨텍스트를 생성하여 이를 서비스 제공기에 전달한다.

3-1 컨텍스트 전처리

컨텍스트 전처리기는 컨텍스트 관리기 내부에서 사용될 수 있도록 입력되는 컨텍스트에 대한 전처

리를 수행한다. 그림 3 과 같이 컨텍스트 전처리기는 통합 텍스트 처리, 조건 컨텍스트 처리, 최종 컨텍스트 처리를 한다. 조건 컨텍스트는 사용자와 서비스로부터 입력되며 각각의 사용자 및 서비스의 해쉬테이블을 통해 관리된다. 통합 컨텍스트는 조건 컨텍스트와 비교하여 일치되면 사용자 컨텍스트로 변환된다. 이때 통합 컨텍스트는 먼저 사용자의 조건 컨텍스트와 비교가 이루어진다. 그리고 해당 사용자의 조건 컨텍스트가 일치하지 않은 경우 다시 서비스의 조건 컨텍스트와 비교하게 된다. 따라서 매치된 사용자의 컨텍스트는 조건 컨텍스트의 What 항목과 통합컨텍스트에서 What 을 제외한 나머지 4W1H 로 부터 생성된다. 최종 컨텍스트는 등록된 서비스와 관련이 있는 컨텍스트를 대상으로 선택적으로 데이터베이스에 추가된다.

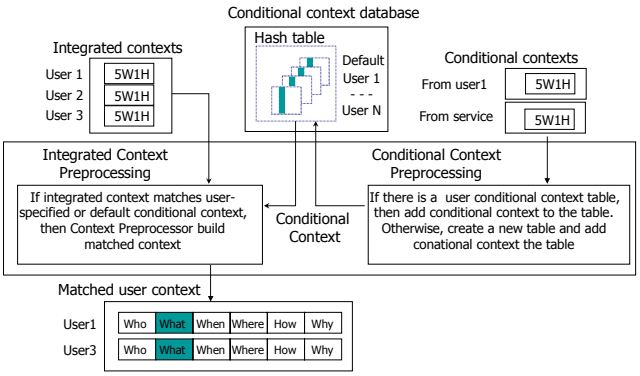

그림 3. 컨텍스트 전처리

3-2 컨텍스트 데이터베이스

컨텍스트 데이터베이스는 서비스 환경 내의 서비스 상황을 유지하기 위해 사용자 컨텍스트, 다른 서비스의 컨텍스트, 사용자의 조건 컨텍스트를 관리한다. 사용자의 컨텍스트는 현재 서비스와 관련되어 있는 사용자의 상황을 유지하며 서비스 이용자가 서비스 환경을 벗어나는 경우 다음 우선순위를 갖는 사용자를 선택하기 위해 사용된다. 다른 서비스의 최종 컨텍스트는 서비스가 실행되기 위한 최종 컨텍스트를 서비스에 전달하기 전에 다른 서비스와의 충돌 관계를 확인하기 위해 사용된다. 그리고 조건 컨텍스트는 사용자에게 제공되는 서비스의 조건들을 관리하며 서비스 환경에서 얻어진 사용자의 컨텍스트와 일치하는지 비교하는데 이용된다.

3-3 최종 컨텍스트 생성기

최종 컨텍스트 생성기는 사용자 간의 충돌이 해결된 컨텍스트를 기반으로 응용 서비스가 활용할 수 있는 최종 컨텍스트를 생성하고 이를 서비스 제공기에 전달한다. 최종 컨텍스트는 서비스 관리기가 제공한 서비스 고유정보와 사용자가 원하는 서비스 동작, 그리고 사용자의 정보를 토대로 생성된다. 그리고 생성된 최종 컨텍스트는 서비스 관리기에 전달된다. 이후, 최종 컨텍스트 생성기는 전달한 최종 컨텍스트가 서비스 제공기에 반영되었으면 이를 다른 서비스에 알리기 위해 네트웍 자가 구성기에게 이 최종 컨텍스트를 전달한다. 반면, 서비스 관리기에서 최종 컨텍스트가 반영되지 않았으면 최종컨텍스트를 다시 서비스 제공기에게 전달한다.

4. 사용자 간의 서비스 충돌 관리

4-1 사용자간 충돌 관리

사용자 간의 충돌 관리기는 서비스 영역 내에서 서비스를 이용하는 사용자 간의 충돌을 해결한다. 사용자 간의 충돌 해결은 사용자의 컨텍스트에 따라 부여되는 우선순위를 기반으로 우선 순위가 높은 사용자를 선택함으로써 이루어진다. 그림 4 는 사용자 충돌 관리기를 나타낸다.

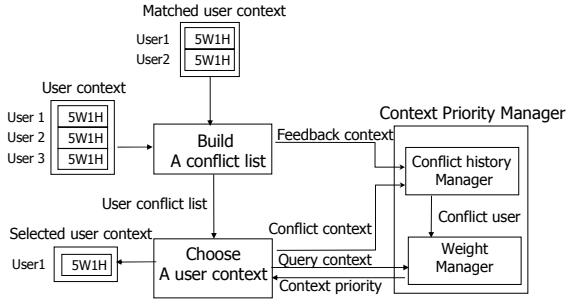

그림 4. 사용자 간 서비스 충돌 관리기

그림 4 와 같이, 사용자 간의 충돌 관리기는 충돌 목록 생성과 사용자 선택의 두 단계로 사용자 간의 충돌을 해결한다. 사용자 충돌목록 생성에서는 서비스를 이용하고자 하는 사용자의 컨텍스트와 기존에 서비스를 사용하고 있는 사용자 중에서 충돌이 예상되는 사용자들의 컨텍스트들을 토대로 충돌 목록을 만든다. 이와함께 사용자의 컨텍스트가 서비스를 직접 제어하거나 서비스를 영역을 벗어나는 경우처럼 사용자의 행동이 서비스에 직접적으로 영향을 미치는 경우 사용자의 피드백으로 간주하고 이를 컨텍스트 우선순위 관리기에 전달한다. 그리고 사용자 컨텍스트 선택 단계에서는 생성된 충돌 목록에서 우선 순위가 높은 사용자를 선택한다. 사용자들에게 부여되는 우선 순위는 컨텍스트 우선 순위 관리기를 통해 얻어진다. 이후 선택된 사용자의 컨텍스트와 서비스를 이용하고 있는 사용자의 컨텍스트는 컨텍스트 우선순위 관리기에 전달된다. 마지막으로 서비스 충돌 관리기는 선택된 사용자의 컨텍스트를 최종 컨텍스트 관리기에 전달한다.

4-2 컨텍스트 우선순위 관리

컨텍스트 우선순위 관리기는 사용자의 충돌 컨텍스트들에 대해 우선순위를 부여한다. 또한 사용자의 컨텍스트에 우선순위를 동적으로 부여하기 위해 사용자들의 충돌 히스토리를 활용하여 컨텍스트의 각 특징들에 대한 가중치를 조절한다. 그림 5 는 컨텍스트 우선순위 관리기를 나타낸다.

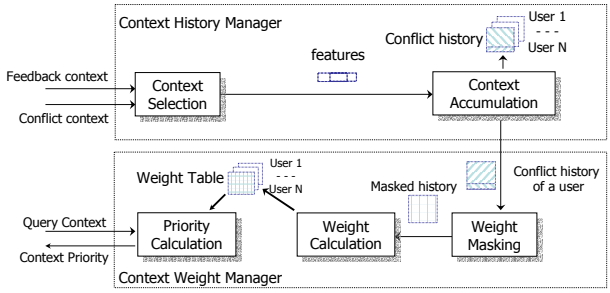

그림 5. 컨텍스트 우선순위 관리기

그림 5 와 같이, 컨텍스트 우선순위 관리기는 히스토리 관리기와 가중치 관리기로 구성된다. 히스토리 관리기는 사용자들의 충돌 컨텍스트를 관리한다. 이를 위해 히스토리 관리기는 사용자 충돌 관리기로부터 사용자의 피드백과 충돌 컨텍스트를 입력받는다. 그리고 이 두 컨텍스트를 통해 충돌 상황에 대한 컨텍스트 특징을 생성한다. 표 1 은 충돌 컨텍스트에 대한 각 특징들을 나타낸다. 이후, 생성된 컨텍스트의 특징은 일정 기간동안 관리될 수 있도록 히스토리 파일에 저장된다. 그리고 히스토리 관리기는 충돌 사용자의 가중치를 조절하기 위해 히스토리 파일로부터 해당 사용자의 충돌 히스토리만을 읽고 이를 컨텍스트 가중치 관리기에 전달한다.

표 1. 충돌 컨텍스트에 대한 특징

Features		Category
What	Service[SVC]	TV: 1 Internet: 2 Movie: 3 Music: 4
Where	Location[L]	Near: 1 Entrance: 2 Couch: 3
When	Time[T]	Hour: 1~24
How	Gesture[G]	Normal: 1 Sit-down: 2 Standup: 3 Moving: 4
Why	Intention[I]	Direct: 1 Attention: 2 Indirect: 3
Who2	Who_Conflict[WC]	ID
Target class	Service change [SC]	No change: 1 Change: 2

컨텍스트 가중치 관리기는 사용자의 우선순위 계산에 이용되는 컨텍스트의 가중치를 관리한다. 또한 변동이 있는 사용자의 히스토리에 대해 컨텍스트의 가중치를 다시 계산한다. 이를 위해 컨텍스트 가중치는 히스토리 관리기로부터 입력되는 사용자의 히스토리에 베이지안 이론을 적용하여 가중치를 계산한다. 식(1)은 베이지안 이론을 나타낸다. 충돌 컨텍스트 X는 $(x_1, x_2, x_3, x_4, x_5, x_6)$로 구성되며 테이블 1 의 SVC, L, T, G, I, WC 의 특징을 각각 나타낸다. 그리고 충돌결과 H 는 (H_1, H_2)이며 Target class 인 SC 를 나타낸다. 따라서, 충돌이 발생 했을 때 현재 서비스를 이용하는 사용자가 서비스를 계속 이용할 확률 $P(H_1|X)$는 사후 확률 $(X|H_1)$와 사전 확률 $P(H_1)$의 곱으로 나타난다.

$$P(H_j \mid X) = \frac{P(X \mid H_j)P(H_j)}{P(X)} \qquad (1)$$

특히, 사용자 간의 충돌에서는 $P(H_1|X)>P(H_2|X)$인 경우 현재 사용자가 서비스를 계속 이용하게 되고 그렇지 않은 경우 다른 사용자가 서비스를 이용하게 된다. 따라서, 컨텍스트의 우선순위는 $P(X|H_1)(H_1)$과 $P(X|H_2)(H_2)$의 최대가 되는 사후 확률의 차를 통해 얻어진다. 이 식에 따라 컨텍스트 각 특징의 가중치는 각 특징의 사전 확률인 $P(x_k|H_j)=s_{kj}/s_j$ 로 표현된다. 여기에서 s_{kj} 는 충돌 컨텍스트가 H_j 에 속하면서 s_k 의 특정한 값을 갖는 컨텍스트의 수이며, s_k 는 H_j 에 속하는 전체 충돌 컨텍스트의 수이다. 가중치 관리기는 이를 기반으로 컨텍스트의 각 특징에 대한 가중치를 계산한다. 그리고 계산된 결과는 추후 검색을 위해 헤쉬테이블과 가중치 파일에 반영된다.

그리고, 가중치 관리기는 충돌관리기가 충돌 컨텍스트 대한 우선순위 값을 요구하면 컨텍스트 가중치 테이블을 기반으로 우선 순위 값을 제공한다. 이를 위해 가중치 관리기는 헤쉬테이블로부터 해당 사용자의 가중치를 가져온다. 이후, 가중치 관리기는 충돌 컨텍스트에 사용자의 가중치를 적용한다. 그리고 식(2)를 적용하여 사후 확률을 계산한다. 사후 확률의 계산은 현재의 사용자가 서비스를 계속 이용할 확률 $P(X_i|H_1)$과 다른 사용자가 이용할 확률 $P(X_i|H_2)$에 대해 이루어진다.

$$P(X \mid H_j) = \prod_{k=1}^{n} P(x_k \mid H_j) \qquad (2)$$

마지막으로, 가중치 관리기는 컨텍스트에 대한 우선 순위는 식(3)과 같이 계산한다. $P(X_i|H_1)(H_1)$는 다른 사용자와의 충돌에서 현재의 사용자가 서비스를 계속 이용할 최대 확률이며 $P(X_i|H_2)(H_2)$는 다른 사용자가 서비스를 이용할 최대 확률이다. 가중치 관리기는 이 두 확률의 차를 충돌 컨텍스트의 우선순위 값으로서 충돌 관리기에게 전달한다.

$$Priority\ (X_i) = P(X_i|H_1)(H_1) - P(X_i|H_2)(H_2) \qquad (3)$$

따라서, 컨텍스트 우선순위 관리기는 사용자의 충돌 히스토리를 바탕으로 컨텍스트에 대한 가중치를 계산하며, 충돌이 발생했을 때 컨텍스트 가중치를 기반으로 사용자의 컨텍스트에 우선 순위를 부여한다.

5. 실험 및 결과

5-1 실험 환경

제안된 컨텍스트 관리기는 다양한 서비스에 활용될 수 있도록 J2SDK 1.4™로 구현되었다. 또한 제안된 컨텍스트 관리기의 유용함을 보이기 위해 스마트 홈 테스트베드인 유비홈에 적용하였다 [5].

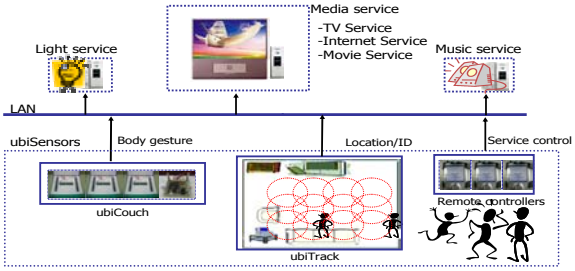

그림 6. ubiTV Service

그림 6 과 같이 유비홈에는 다양한 응용서비스와 센서들이 ubi-UCAM 을 기반으로 구현되어 있다 [4]. 특히, 제안된 컨텍스트 관리기의 유용함을 검증하기 해 음악, 영화 등 다양한 미디어 서비스들이 활용되었다. 또한, 이와 함께 사용된 센서로는 사용자의 위치를 추적하는 ubiTrack, 사용자의 움직임을 감지하는 소파센서, 그리고 서비스 제어를 위해 PDA 리모컨이 사용되었다. 그리고 사용자는 일반적인 가정의 가족 구성원인 4 인 가족으로 설정하였다.

5-2 실험결과 및 분석

사용자 간의 충돌을 컨텍스트를 활용하여 해결하기 위해 다음과 같은 실험을 하였다. 사용자는 가족 구성원 중에서 서비스 선호도가 서로 다른 아버지와 아들로 선정하였다. 그리고 이들은 홈 환경에서 시간에 따라 자신이 선호하는 서비스를 제공 받게 된다. 표 2 는 텔레비젼 서비스에 대한 사용자의 컨텍스트 가중치를 나타낸다

표 2. 충돌 컨텍스트의 가중치

User	What (Service)	Where (Location)	When (Time)	How (Behavior)
Father (0.64)	Type1:0.33 Type2:0.43 Type3:0.33 Type4:0.38	Near: 0.55 Sofa: 0.83 Other:0.71	19: 0.1 20: 0.83 21: 0.91 22: 0.5 23: 0.7	Sit-down: 0.85 Moving: 0.63 In:0.45
Son (0.51)	Type1:0.9 Type2:0.5 Type3:0.33 Type4:0.57	Near:0.5 Sofa:0.87 Other:0.11	19: 0.9 20: 0.8 21: 0.1 22: 0.4 23: 0.5	Sit-down: 0.9 Moving: 0.59 In: 0.11

표 2 과 같이, 아들의 경우 코믹이나 드라마처럼 호기심을 자극하는 프로그램을 선호하기 때문에 19 시 20 시에 서비스를 많이 사용하는 것으로 나타났다. 반면, 아버지의 경우 스트레스 해소나 정보를 얻기 위해 시청하기 때문에 뉴스나 시사에 대한 프로그램이 방송되는 21 시와 23 시에 높은 것으로 나타났다. 따라서 텔레비젼 서비스는 시간과 사용자의 선호도에 따라 우선 순위가 결정됨을 알 수 있다.

그리고 제안된 충돌해결 방법의 정확도를 측정하기 위해 사용자 간의 충돌 상황을 제안된 방법과 "고정된 우선순위" 규칙을 함께 실험을 하였다. 그림 7 은 두 가지 충돌 해결 방밥을 사용자 간의 서비스 충돌해결에 적용한 결과를 나타내고 있다.

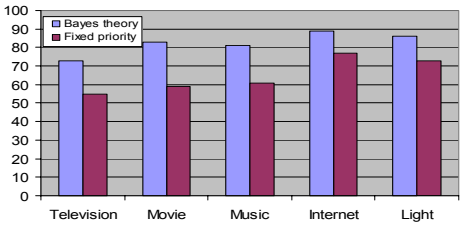

그림 7. 사용자 선택방법에 따른 예측율

그림 7 과 같이, 인터넷 서비스나 전등서비스 처럼 서비스에 대한 특별한 선호도를 나타내지 않은 경우에는 두 가지 충돌해결 방법의 차이가 적게 나타났다. 반면 텔레비젼 서비스, 영화 서비스, 음악 서비스에서는 많은 차이가 발생함 알 수 있다. 이러한 차이는 서비스의 내용과 상황에 따라 사용자의 선호도가 변하기 때문이다. 이러한 경우 고정된 우선순위를 가지는 해결 방법은 이러한 변화를 반영하기 힘들다. 반면, 사용자의 히스토리를 기반으로 통계적으로 접근하는 제안된 방법의 경우 사용자의 선호도에 따라 우선순위를 부여하여 사용자 간의 충돌을 해결하였다.

5. 결 론

본 논문에서는 유비쿼터스 컴퓨팅 환경에서 다중 사용자들이 상황인지 응용서비스를 이용하는 경우 발생되는 사용자 간의 서비스 충돌을 해결하기 위한 컨텍스트 관리기를 제안하였다. 제안된 컨텍스트 관리기는 사용자 간의 충돌이 발생했을 때 컨텍스트를 활용하여 우선 순위에 따라 사용자를 선택하였다. 또한 사용자의 피드백을 통해 사용자의 성향을 파악하여 사용자에 맞게 우선순위가 조절되도록 하였다. 추후 연구로는 사용자 간의 충돌을 다양하게 관찰하기 위해 더 많은 사용자를 대상으로 충돌 실험을 진행할 예정이다. 또한, 이들의 서비스 이용 형태를 장시간 동안 진행하여 사용자들의 우선순위 변화를 관찰할 계획이다.

6. 참고 문헌

[1] Anind K. Dey, "Understanding and Using Context. Personal and Ubiquitous Computing, Special issue on Situated Interaction and Ubiquitous Computing, 5(1), 2001

[2] Anind K. Dey and Gregory D. Abowd, The Context Toolkit: Aiding the Development of Context-Aware Applications, Proceedings of the Workshop on Software Engineering for Wearable and Pervasive Computing (SEWPC), Limerick, Ireland, June 6, 2000.

[3] Nicholas Hanssens, Ajay Kulkarni, Rattapoom Tuchinda, and Tyler Horton, "Building Agent-Based Intelligent Workspaces," In ABA Conference Proceedings, June 2002.

[4] S.Jang, and W.Woo, "ubi-UCAM: A Unified Context-Aware Application Model", Lecture Note Artificial Intelligence, Vol, 2680, pp.178-189, 2003

[5] Y.Oh, W.Woo, "A unified Application Service Model for ubiHome by Exploiting Intelligent Context-Awareness," UCS2004.