

상황 정보 온톨로지 기반 추론 검증 도구

(An Inference Verification Tool based on a Context Information Ontology)

김 목 련[†] 박 영 호^{**}
(Mokryun Kim) (Youngho Park)

요 약 유비쿼터스 환경에서 모바일 단말기의 제한적인 자원 문제를 해결하기 위해 주변 자원을 실시간으로 공유하는 연구들이 진행되고 있다. 그리고 자원의 공유뿐만 아니라 상황 정보에 기반한 추론을 통해 개인 맞춤형 자원을 추천하는 연구도 활발히 진행되고 있다. 개인 맞춤형 자원 추천을 위하여 사용자의 기본 정보, 자원에 대한 선호도, 공유 대상이 되는 자원의 정보, 단말기의 위치, 시간과 같은 다양한 상황 정보는 효과적으로 공유 및 관리되어야 한다. 또한 신뢰성 있는 자원 추천을 위해 필요한 추론규칙을 검증하는 단계는 매우 중요하다. 이를 위해서 다양한 상황 정보를 구성하여 실제 단말기 상에서 자원 추천규칙이 올바르게 동작하는지 검증해야 하지만 이는 현실적으로 많은 비용과 시간이 필요하다. 따라서 본 논문에서는 이러한 문제점을 해결하기 위하여 추론 검증 도구를 제안한다. 제안하는 추론 검증도구는 편리한 그래픽 사용자 인터페이스를 제공하여 원하는 상황 정보를 쉽게 생성할 수 있고, 실제 단말기를 대신하여 동적인 상황 정보의 변경에 따른 추론을 정확하게 검증한다.

키워드 : 추론 검증, 자원 공유, UMO, 온톨로지, 유비쿼터스

Abstract In ubiquitous environments, invisible devices and software are connected to one another to provide convenient services to users. In order to provide such services, we must have mobile devices that connect users and services. But such services are usually limited to those served on a single mobile device. To resolve the resource limitation problem of mobile devices, a nearby resource sharing research has been studied. Also, not only the nearby resource share but also a resource recommendation through context-based resource reasoning has been studied such as an UMO Project. The UMO Project share and manage the various context information for the personalization resource recommendation and reason based on current context information. Also, should verify resource inference rules for reliable the resource recommendation. But, to create various context information requires huge cost and time in actuality. Thus, we propose a inference verification tool called USim to resolve problem. The proposed inference verification tool provides convenient graphic user interfaces and it easily creates context information. The USim exactly verifies new inference rules through dynamic changes of context information.

Key words : inference verification, resource share, UMO, Ontology, Ubiquitous

· 본 연구는 2009년도 정부(교육과학기술부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임(No. 2-0905-0012)

† 학생회원 : 숙명여자대학교 멀티미디어학과
mrkim@sm.ac.kr

** 종신회원 : 숙명여자대학교 멀티미디어학과 교수
yhpark@sm.ac.kr

논문접수 : 2008년 12월 15일

심사완료 : 2009년 8월 16일

Copyright©2009 한국정보과학회 : 개인 목적이나 교육 목적인 경우, 이 저작물의 전체 또는 일부에 대한 복사본 혹은 디지털 사본의 제작을 허가합니다. 이 때, 사본은 상업적 수단으로 사용할 수 없으며 첫 페이지에 본 문구와 출처를 반드시 명시해야 합니다. 이 외의 목적으로 복제, 배포, 출판, 전송 등 모든 유형의 사용행위를 하는 경우에 대하여는 사전에 허가를 얻고 비용을 지불해야 합니다.

정보과학회논문지 : 시스템 및 이론 제36권 제6호(2009.12)

1. 서 론

유비쿼터스 컴퓨팅 환경 속에서 사용자는 시간과 장소에 구애 받지 않고 자유롭게 컴퓨팅 환경에 접속할 수 있고, 유비쿼터스 컴퓨팅 서비스는 사용자의 요구를 스스로 예측, 인지하여 사용자에게 필요한 서비스를 하게 된다[1].

유비쿼터스 컴퓨팅 환경에서 다양한 서비스를 제공하기 위해서 사용자와 서비스 간에 매개체 역할을 하는 모바일 단말기가 필수적이다. 하지만 자원이 제한적인 모바일 단말기의 특성상 다양한 기능을 이용할 수 없다.

따라서 모바일 단말기의 제한적인 자원의 문제를 해결하기 위하여 주변의 자원을 실시간으로 공유하는 연구가 많이 진행되고 있다. 그리고 주변 자원의 공유뿐만 아니라 모바일 단말기의 사용자 프로파일과 자원에 대한 선호도를 분석하여 개인 맞춤형 자원을 추천해 주기 위한 Celadon[2], CoBrA(Context Broker Architecture)[3], GIAI[4], UMO(Ubiquitous Mobile Object) 소프트웨어 프레임워크[5,6]와 같은 연구들도 활발히 진행되고 있다.

그 중에서 UMO(Ubiquitous Mobile Object) 소프트웨어 프레임워크[5,6]는 상황 인지 기반의 개인화 된 자원 추론 및 공유 소프트웨어를 설계한 연구로 주변의 자원을 실시간으로 공유함으로써 모바일 단말기의 제한적인 자원 문제를 해결한다. UMO(Ubiquitous Mobile Object)는 앞서 설명한 개인화 된 자원 추론 및 공유 시스템이 내장되어 자원 추론 및 공유를 제공할 수 있는 모바일 단말기이다. 즉, UMO는 변화하는 사용자의 자원에 대한 선호도 및 자원, 환경과 같은 상황 정보에 기반을 둔 추론을 통해 개인 맞춤형 자원을 추천한다. UMO의 자원 추천 및 공유 사례는 다음과 같다.

사례. 오전 10시에 A라는 사람이 작은 디스플레이를 가진 UMO로 동영상을 보면서 이동 중이다. 그러던 중 A는 서로 다른 속성을 가 진5개의 모니터 자원이 있는 회의실에 도착한다. A는 더 효과적으로 동영상을 감상하기 위하여 자신의 UMO에 ‘동영상 재생’ 서비스를 요청한다. 현재 시간이 오전 10시이고 A가 보유한 UMO의 위치가 위도 200, 경도 200, A가 1024*768의 해상도와 17인치 크기를 갖는 모니터를 선호한다. UMO는 먼저 시간과 위치 정보를 고려하여 적합한 모니터를 추론한다. 현재 시간이 오전이기 때문에 화면 밝기가 높은 모니터가 추론된다. 그리고 UMO의 위치를 고려 하여 UMO와 거리가 먼 모니터는 추론 대상에서 제외된다. 마지막으로 자원 선호도를 고려한다. 추론된 모니터 중에서 사용자의 선호도가 더 일치하는 모니터를 우선순위로 정렬하여 사용자에게 추천한다. 사용자는 추천된 자원들 중, 실제로 사용할 자원을 선택 한다. 선택된 모니터는 실제로 이용할 수 있도록 필요한 드라이버를 설치하고 동영상 데이터를 전달하여 재생한다. □

위의 사례와 같이, 사용자의 자원에 대한 선호도와 현재 시간, 위치 정보를 고려하여 수집된 자원들 중 최적의 자원을 사용자에게 추천한다. 이렇게 자원 추천을 위해서 다양한 자원에 대한 정보뿐만 아니라, 사용자 개인 정보, 자원에 대한 선호도 정보, 현재 시간, 위치와 같은 상황 정보를 미리 수집하여 유지 및 공유해야 한다. 또한 다양하게 변경되는 상황 정보에 기반을 두어 정확성 있는 자원 추천이 이루어지는지 반복적으로 시험 및 평

가해야 한다. 하지만 이는 현실적으로 불가능 하다. 그 이유는 첫째, 자원의 종류가 매우 다양함으로 이를 모두 구비하여 추론을 위한 주변 환경을 구성하기에 많은 비용이 필요하다. 둘째, 자원을 구비할 비용이 있다 하더라도 섬세하게 자원을 제어할 수 없다. 셋째, 개발 중인 새로운 자원에 대한 실험이 불가능하다. 마지막으로 사용자의 자원에 대한 선호도와 시간 및 위치 정보를 수동으로 다양하게 생성 및 조합하기 위해서 많은 시간이 필요하다.

위와 같은 문제점을 해결하기 위하여 존재하지 않는 자원이거나 실현 불가능한 상황 정보와 같은 가상의 다양한 상황 정보를 생성하여 추론 규칙을 검증할 수 있는 USim(UMO Simulator)를 제안한다. USim은 로컬 USim과 외부 USim으로 나누어 구현한다. 로컬 USim은 서비스 요청과 사용자의 자원 선호도, 위치 및 시간, 소유 자원 정보를 생성 및 변경한다. 외부 USim은 주변 UMO 및 주변 UMO의 보유 자원 및 위치 정보를 생성 및 변경한다. USim은 정의한 온톨로지에 기반하여 정보를 생성하고 편리한 그래픽 사용자 인터페이스를 제공하여 생성한 상황 정보의 수정, 삭제, 상호 관계 설정을 쉽게 할 수 있도록 지원한다.

본 논문의 다음과 같은 새로운 공헌을 제시한다.

- 그래픽 사용자 인터페이스를 제공하여 가상의 상황 및 자원 정보를 편리하게 생성할 수 있는 USim을 설계 및 개발
- USim을 통해 상황 정보를 고려한 자원 추론 규칙을 검증할 수 있는 방안을 제시

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구로서 상황 인지 기반 자원 공유 서비스인 IBM의 Celadon, UMBC의 CoBrA, UMO 프로젝트에 대해 기술한다. 3장에서는 USim의 설계 및 구현에 대해 설명한다. 그리고 4장에서는 USim을 검증한다. 마지막으로 5장에서는 본 논문의 결론을 내린다.

2. 관련 연구

본 장은 관련 연구로서, 상황 정보 기반으로 자원 공유를 지원하는 선행 연구에 대해 기술한다.

2.1 Celadon

Celadon[2]은 상황 인식을 통해 모바일 단말기 간의 기능적인 협업을 제공하는 주는 브로커 중심의 미들웨어 인프라를 구축하는 연구이다. 이를 위해서 동적 탐색 기술, 상황인식 기술, 디바이스 간 통신 기술, 디바이스 간 협업 기술, 서비스 조합기술과 같은 연구를 수행 중이다. 그림 1은 Celadon의 인프라 설계 구조이다. Zone Collaboration Broker에서 분산된 Zone들을 관리하고, Zone Service Broker에서 Zone에서 제공하는 서비스들

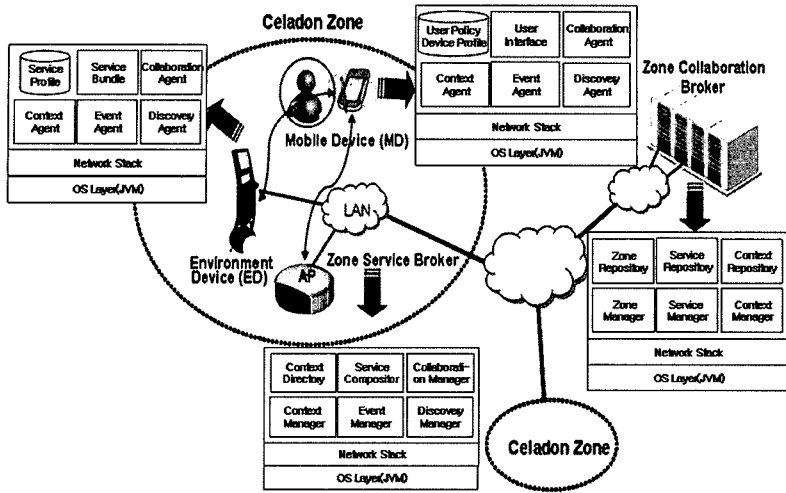


그림 1 Celadon 인프라 설계 구조[2]

관리한다. Zone내의 Environment Device는 서비스를 실제적으로 사용자 디바이스에 전송해 주기 위한 모듈을 가지고 있다. 사용자의 모바일 디바이스에는 사용자의 프로파일 정보와 서비스를 받아 수행할 수 있는 모듈을 가지고 있다.

2.2 CoBrA

UMBC(University of Maryland, Baltimore Country)에서 개발된 CoBrA는 지능형 공간상에 존재하는 모든 컴퓨팅 엔티티들을 위한 상황 공유 모델을 관리하는 컨텍스트 브로커를 개발하는 연구이다[3]. 지능형 공간에서 사용되는 상황 정보를 온톨로지로 구성하여 상황에

대한 기본적인 추론이 가능하도록 하였다. 그림 2는 컨텍스트 브로커를 중심으로 한 CoBrA의 구조이다. 컨텍스트 브로커를 중심으로 Information Server, Semantic Web & Web Services, Database와 같은 외부 자원에서의 컨텍스트와 지능형 공간에서 Smart Tag Sensor, Environment Sensors, Device & Gadget Sensor를 통해 수집된 컨텍스트를 저장하고, 추론한다. 그리고 주변에 여러 에이전트와 상환 인지 자원들과의 표준화된 통신을 통하여 수집된 컨텍스트를 활용한다. 이처럼 CoBrA는 컨텍스트 브로커를 중심으로 중앙화된 상황 관리 모델을 구성하여 상황 정보를 관리한다. 하지만 이러한 중

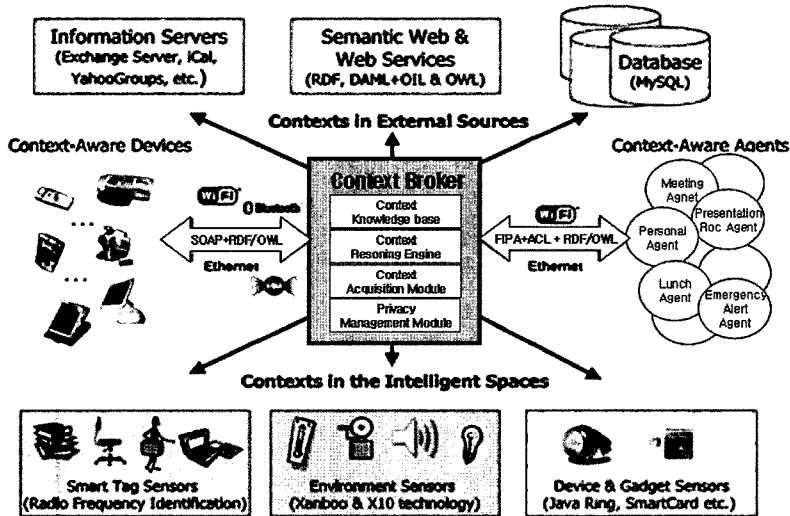


그림 2 CoBrA의 컨텍스트 브로커 구조[3]

양 집중식의 데이터 처리는 이동형 환경에서 적용하기에 매우 취약하다는 단점이 있다.

CoBrA는 사용자의 의도에 부합하는 서비스를 제공하기 위하여 주변 자원들을 공유하여 사용한다. 하지만 UMO 소프트웨어 프레임워크에서는 동적으로 변경되는 상황 정보와 사용자의 자원 선호도에 기반하여 다수의 자원들 중에서 가장 적합한 자원을 추론하여 사용자에게 선호도에 기반을 둔 자원 우선순위를 결정하여 추천하는 서비스를 제공한다. 즉, 사용자의 의도에 맞는 주변 자원을 공유한다는 점은 동일하지만, UMO 소프트웨어 프레임워크에서는 공유되는 자원이 물리적 상황 및 사용자의 선호도에 고려하여 사용자에게 가장 적합한 자원을 공유한다는 점이 다르다.

2.3 UMO 프로젝트

UMO 프로젝트는 자원 추론 엔진이 내장되어 유비쿼터스 컴퓨팅 서비스를 제공하는 모바일 단말기인 UMO 개발을 목표로 한다. 현재는 UMPC(Ubiquitous Mobile Personal Computer)와 같은 휴대 단말기나 소형 메달 모양의 휴대기기 형태로 UMO가 개발되었다. UMO는 주위의 노트북이나 프린터, 무선모뎀, 저장장치, 키보드, 마우스 등 이용할 수 있는 모든 컴퓨팅 장치들을 지그비, 무선랜, 블루투스, CDMA 등 무선 네트워크로 자동 인식해 사용자가 하고자 하는 작업을 빠르게 연계시킨다.

UMO 개발을 위한 연구 분야는 크게 UMO 소프트웨어 프레임워크 개발과 하드웨어 프레임워크 개발로 나뉘어진다. UMO 소프트웨어 프레임워크는 주변 자원 공유 및 자원 추론, 추천을 담당하고, 하드웨어 프레임워크는 UMO의 프로세서, 메모리, 통신 모듈을 담당한다.

앞서 언급한 연구들은 이동 에이전트 기반이거나 클라이언트-서버 기반이다. UMO 프로젝트 [5,6]는 모바일 단말기 간의 Peer to Peer 형식으로 작동한다. 따라서 별도의 서버나 에이전트의 구축 없이 서비스가 가능하다.

UMO 소프트웨어 프레임워크는 그림 3과 같이 크게

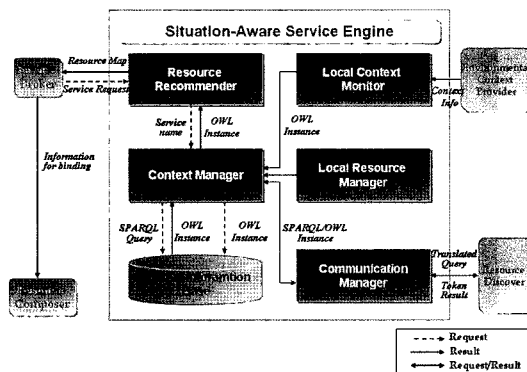


그림 3 UMO 소프트웨어 프레임워크

Service Broker, Situation-Aware Service Engine, Resource Composer, Resource Discoverer. Environmental Context Provider로 구성된다[6].

본 논문에서는 UMO 소프트웨어 프레임워크에서 자원 추론 및 추천을 담당하는 상황 인지 서비스 엔진(Situation-Aware Service Engine)에서 Context Monitor와 Resource Manager, Communication Manager 대신하여 가상의 상황 정보를 생성하고, 이를 바탕으로 추론 및 추천을 편리하게 검증할 수 있는 USim을 개발한다.

3. USim(UMO Simulator)

본 절에서는 먼저 제안하는 USim의 필요성에 대한 이해를 돕기 위하여 UMO 소프트웨어 프레임워크의 상황 인지 서비스 엔진에 대해 구체적으로 설명 한다. 그런 후 USim의 구조, 주요 알고리즘, 구현에 대해 구체적으로 기술한다.

3.1 UMO 상황 인지 서비스 엔진

그림 4는 그림 3의 상황 기반 서비스 엔진의 상세 구성도이다. 상황기반 서비스 엔진은 크게 6개의 모듈로 구성된다.

Local Context Monitor(LCM)는 사용자의 자원에 대한 선호도, 사용자 정보를 User Profile Configuration을 통해 수집하고, 모바일 단말기의 위치 및 현재의 시간과 같은 환경 정보를 Local Context Aggregator를 통해 수집한다. User Context Collector는 수집된 정보를 최종적으로 Context Information Base(CIB)에 저장할 수 있는 온톨로지 인스턴스로 변환하여 Context Manager(CM)를 통해 CIB에 저장한다.

Local Resource Manager(LRM)는 현재의 모바일 단말기가 소유하고 있는 가용한 자원 정보를 수집하는 역할을 수행한다. Resource Change Detector에서 가용한 자원을 수집하고, Resource Configuration 모듈은 수집된 자원 정보를 CIB에 저장할 수 있는 온톨로지 인스턴스로 변환하여 CM을 통해 CIB에 저장한다.

Context Manager(CM)는 다른 모듈로부터 전달되는 정보를 CIB에 전달하는 중간 매개체 역할을 수행한다. 그리고 SPARQL Generator는 CIB에 저장된 정보를 검색하기 위한 SPARQL질의를 생성하는 모듈이다.

Communication Manager(CCM)는 사용자가 요청한 서비스에 필요한 자원을 수집하기 위해, Resource Discover로 주변 모바일 단말기에 가용한 리소스 정보를 요청하는 질의를 전달하고 그 결과를 받아 CIB로 전달하는 역할을 수행한다. CCM의 Converter모듈에서는 SPARQL질의를 실제 통신에서 전달 가능한 질의의 형태로 변환 역할과 반대로 질의의 포맷을 SPARQL 질의 형태로 변환하는 역할을 수행한다. 그리고, SPARQL질

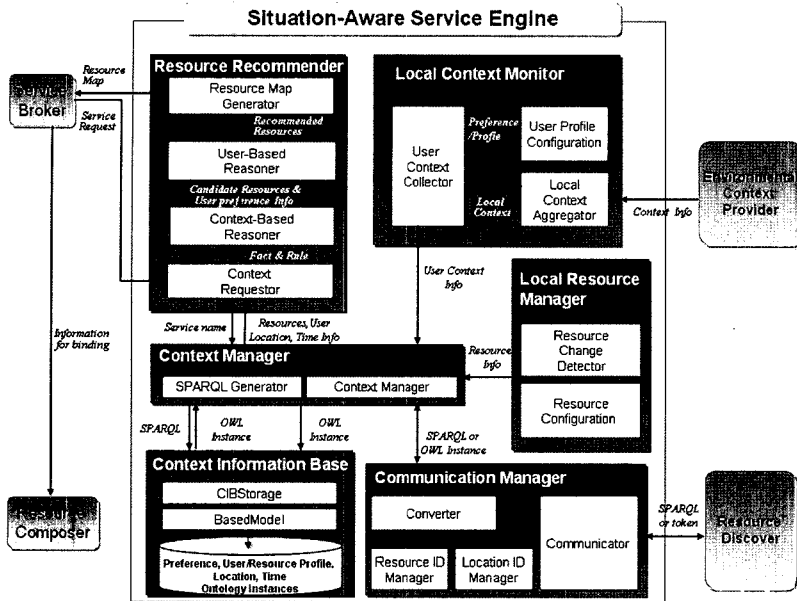


그림 4 상황 인지 서비스 엔진의 상세 구성도

의에 대한 결과인 온톨로지 인스턴스를 실제 통신에서 전달할 때 데이터 량을 줄이기 위하여 자체 정의한 토큰 타입으로 변환해 주는 역할과, 반대로 토큰 타입의 데이터를 CIB에 저장할 수 있도록 다시 온톨로지 인스턴스로 변환하는 역할을 수행한다. Resource ID Manager와 Location ID Manager는 외부 모바일 단말기로부터 수집된 정보 중, 자원을 구분하기 위한 자원 아이디와 로케이션 아이디가 현재 모바일 단말기가 소유한 정보와 중복 될 수 있으므로, 외부 모바일 단말기로부터 전달받은 자원과 로케이션의 아이디를 변경하는 역할을 수행한다.

Context Information Base(CIB)는 본 논문에서 제안하는 상황 정보 온톨로지 관리 모듈로써, 다른 모듈로부터 전달되는 정보들을 저장하고, 다른 모듈로부터 전달되는 정보의 수정 및 삭제 요청을 처리하여 서비스에 필요한 최신의 상황 정보를 유지한다. 그리고 CM로부터 전달되는 SPARQL질의를 수행하여 질의에 맞는 결과를 전달한다. CIB의 온톨로지 설계 및 동작 과정은 4장 상황 정보 온톨로지 관리에서 자세히 설명한다.

마지막으로 Resource Recommender(RR)는 사용자가 요청한 서비스 명을 CM으로 전달하고, 그에 따른 상황 정보들을 전달 받는다. 전달받은 상황 정보의 추론을 통해 사용자에게 맞는 자원을 추천한다.

사용자가 서비스를 요청하기 위해서는 모바일 단말에 UMO 소프트웨어가 설치되어 있어야 하며, UMO 소프트웨어가 구동되면, 먼저 LCM와 LRM이 모바일 단말

기가 소유하고 있는 가용한 자원 정보 및 사용자 프로파일, 선호도, 위치, 시간 정보를 CM로 전달하여 CIB에 저장한다. 그런 후, 사용자가 서비스를 요청하면 다음과 같은 단계로 상황 인지 서비스 엔진이 동작한다.

단계 1. 사용자의 서비스 요청 이벤트 처리 과정은 사용자가 Service Broker를 통해 '문서편집' 서비스를 요청하게 되면, 서비스 명은 Resource Recommender를 통해 CM로 전달한다(그림 5의 ①).

단계 2. CM은 서비스에 필요한 자원들이 주변 UMO에 존재하는 검색하기 위한 SPARQL 쿼리를 생성하여 CCM으로 전달한다(그림 5의 ②).

단계 3. CCM은 주변 UMO를 탐색한 Resource Discovery로 SPARQL를 전달하고, Resource Discovery는 탐색한 주변 UMO에게 Zigbee 통신을 통해 쿼리를 모두 전달한다(그림 5의 ③).

단계 4. 주변 UMO(그림 5의 UMO2, UMO_n)의 CCM은 전달받은 쿼리를 CIB로 전달하기 위해 CM으로 전달하고(그림 5의 ④), CM은 UMO의 컨텍스트 정보를 유지하고 있는 CIB에 쿼리 전달하여 그 결과를 받는다(그림 5의 ⑤, ⑥).

단계 5. CIB로부터 전달 받는 결과는 UMO가 보유하고 있는 자원 중, 문서 편집에 필요한 자원의 정보와 UMO의 정보, 위치 정보로 구성된 온톨로지 인스턴스이다. SPARQL의 검색 결과를 요청한 UMO1로 전달하기 위하여 CM은 결과를 CCM으로 전달한다(그림 5의 ⑦).

단계 6. CCM은 내부 Converter모듈을 통해 온톨로

지 인스턴스를 단순한 형태의 Token으로 변환하고 Resource Provider로 전달하여, UMO1로 전송한다(그림 5의 ⑧).

단계 7. UMO1은 주변 UMO로부터 전달받은 Token 형태의 결과를 온톨로지 인스턴스로 변환하여 자신의 CIB에 저장한다(그림 5의 ⑨).

단계 8. 주변 UMO의 정보가 함께 저장된 CIB에 서비스에 필요한 각각의 자원 검색을 위한 SPARQL을 수행하고(그림 5의 ⑩), 최종적으로 추론에 필요한 자원 정보, 사용자 선호도, 프로파일, 위치, 시간 정보를 전달 받아 ResourceRecommender로 전달한다(그림 5의 ⑪).

단계 9. Resource Recommender에서는 먼저 위치, 시간정보를 자원 추론 규칙에 적용하여 수집된 자원 중 적절한 자원을 추론한다. 그 다음, 추론된 자원 중에서 사용자의 선호도 정보를 고려하여 선호도에 가장 부합하는 자원 순으로 정렬한 자원맵을 Service Broker를 통해 사용자에게 추천한다.(그림 5의 ⑫).

단계 10. 사용자가 최종적으로 이용할 자원을 선택하면, 선택된 자원의 정보를 Resource Composer로 전달하여(그림 5의 ⑬) 자원을 실제로 이용할 수 있도록 자원을 연결한다.

3.2 USim의 필요성

그림 5와 같이 사용자가 요청한 서비스에 적합한 자원을 추론하기 위해서 자신의 자원뿐만 아니라 주변 UMO들을 탐색하여 필요한 자원을 검색하는 쿼리를 전달하게 된다. 하지만 구성할 수 있는 주변 UMO의 개수에 한계가 있고 UMO의 보유 자원을 실제로 다양하

게 구성하는 것은 현실적으로 불가능 하다. 그 이유는 자원의 종류가 매우 다양함으로 이를 모두 구비하여 하기에는 많은 비용이 필요하다. 만일 자원을 구비할 비용이 있다 하더라도 자유롭게 자원을 제어 할 수 없다. 그리고 개발 중인 새로운 자원에 대한 실험이 불가능하다. 마지막으로 사용자의 자원에 대한 선호도와 시간 및 위치 정보를 수동으로 다양하게 생성 및 조합하기 위해서 많은 시간이 필요하다. 이러한 문제점으로 실제 UMO를 가지고 새롭게 추가된 자원 추론 규칙에 맞게 추론이 올바르게 수행되는지 검증하는 과정도 매우 어렵다.

따라서 이를 해결하기 위해, 다양한 상황 정보를 구성하여 작성한 추론 규칙에 따라 올바른 자원이 추론되는지 검증하는 자원 추론 검증 도구인 USim을 제안한다. USim은 내부 USim과 외부 USim으로 구성된다. 내부 USim은 보유 자원 및 환경, UMO 사용자의 프로파일, 선호도를 다양하게 구성하고, 서비스의 요청 및 자원 추론 및 추천 결과를 나타내는 역할을 수행한다. 외부 USim은 다수의 주변 UMO 정보를 생성하고 UMO의 자원 및 환경 정보를 다양하게 생성, 수정, 삭제할 수 있도록 지원한다. 따라서 USim을 이용하면 사용자에게 맞는 자원이 추천되는지를 지속적으로 상황 정보를 변경하면서 그 결과를 판단할 수 있다. 또한 새로운 추론 규칙에 오류가 있는지 검증할 수 있다.

3.3 USim의 구조

그림 6은 USim의 구조 이다. 그림 6과 같이 USim은 크게 일곱 부분으로 구성된다. 먼저 Resource Simulator는 UMO와 자원의 정보 출력, 추가, 수정, 삭제를 위한

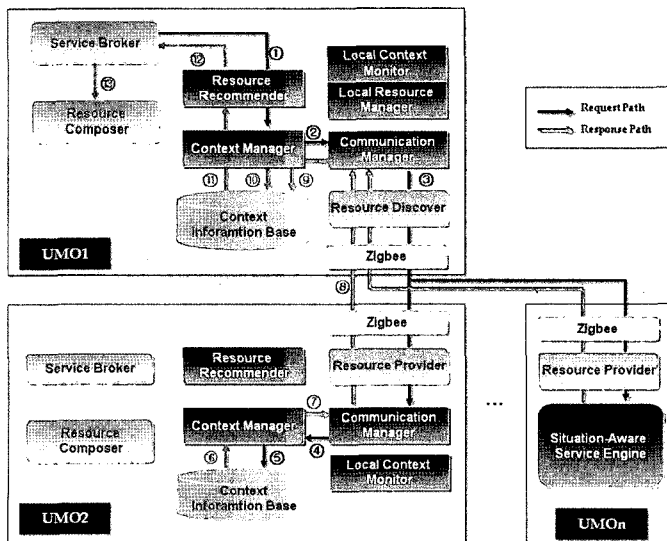


그림 5 서비스 요청 이벤트 처리 과정

GUI(Graphic User Interface)를 제공하고, GUI를 통해 발생하는 정보의 연산 이벤트를 처리하기 위해 CM을 통해 CIB에 저장된 UMO와 자원 정보의 연산을 요청한다.

Context Simulator는 UMO의 위치와 현지 시간 정보의 화면 출력, 추가, 수정, 삭제에 위한 GUI를 제공하고, 역시 GUI를 통해 발생하는 정보의 연산 이벤트를 처리하기 위해 CM을 통해 CIB에 저장된 위치와 시간 정보의 연산을 요청한다.

User Simulator는 UMO의 소유자의 프로파일과 자원의 선호도 정보의 화면 출력, 추가, 수정, 삭제에 위한 GUI를 제공하고, 역시 GUI를 통해 발생하는 정보의 연산 이벤트를 처리하기 위해 CM을 통해 CIB에 저장된 사용자 정보의 연산을 요청한다.

Scenario Creator는 자동적인 상황 정보를 구성하는 역할을 수행하는 모듈으로써, 자동적인 상황 정보 구성의 설정 여부 및 상황 정보 설정 간격을 사용자로부터 입력 받는 GUI를 제공한다. 자동 상황 정보 구성이 설정되면, 입력 받은 설정 간격마다 Context Simulator와 User Simulator, Resource Simulator에서 생성된 각각의 정보들을 조합하여 새로운 가상의 상황 정보를 생성한다.

CM은 다른 모듈로부터 전달되는 정보를 CIB에 전달하는 중간 매개체 역할을 수행하고, CIB에 저장된 정보를 검색하기 위한 SPARQL질의를 생성하는 역할을 수행한다.

CIB는 각각의 USim에서 생성한 상황 정보들을 유지하고 있다. 마지막으로 CCM은 UMO로부터 전달된 질의를 CM으로 전달하여 CIB에 저장되어 있는 각 UMO 정보를 검색하여 그 결과를 질의를 보낸 UMO로 전달하는 역할을 수행한다.

내부 UMO에 존재하는 QuerySimulator는 사용자에게 원하는 서비스를 입력 받고, 서비스를 지원하기 위한

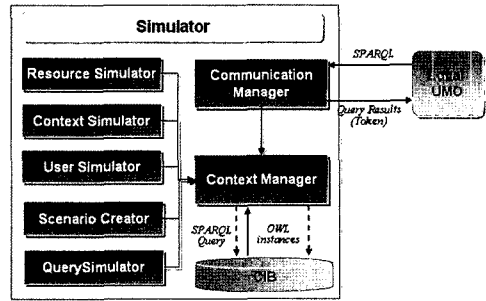


그림 6 USim의 구조

주변 UMO의 자원 검색 결과와 RR로부터 수행된 1차, 2차 추론 결과를 출력하는 역할을 수행한다.

3.4 USim의 구현

본 절에서는 USim에서 제공하는 기능을 구현된 인터페이스 화면과 함께 설명한다. 그림 7은 USim에서 제공하는 기능의 플로우차트이다. USim은 크게 File, Edit, Query, Option, Help 메뉴로 구성된다. File메뉴에서는 수정된 상황 정보를 저장하는 Save와 USim을 종료하는 Exit기능이 있다. Edit에는 UMO, User, Resource, Location, Preference 정보를 추가, 수정, 삭제하는 기능이 있다. Query메뉴에는 문서편집, 프리젠테이션, 동영상 재생과 같은 서비스 요청을 할 수 있고, 그에 따른 자원 검색 및 추론 결과를 출력한다. Option 메뉴에는 자동으로 상황 정보를 구성할 수 있는 Scenario 설정 기능이 있다. Help에는 USim에 대해 소개한다.

USim에서 제공하는 주요 기능을 정리하면 다음과 같다. 상황 정보 온톨로지는 UMO, 자원, 위치, 시간, 사용자 프로파일, 사용자 자원 선호도 정보를 의미한다.

- 상황 정보 온톨로지의 출력
- 새로운 상황 정보 온톨로지의 생성
- 상황 정보 온톨로지의 수정, 삭제
- 상황 정보 온톨로지의 관계 설정

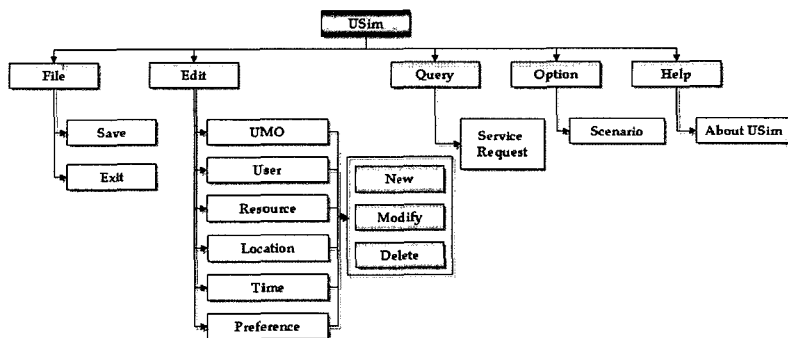


그림 7 USim의 기능 플로우차트

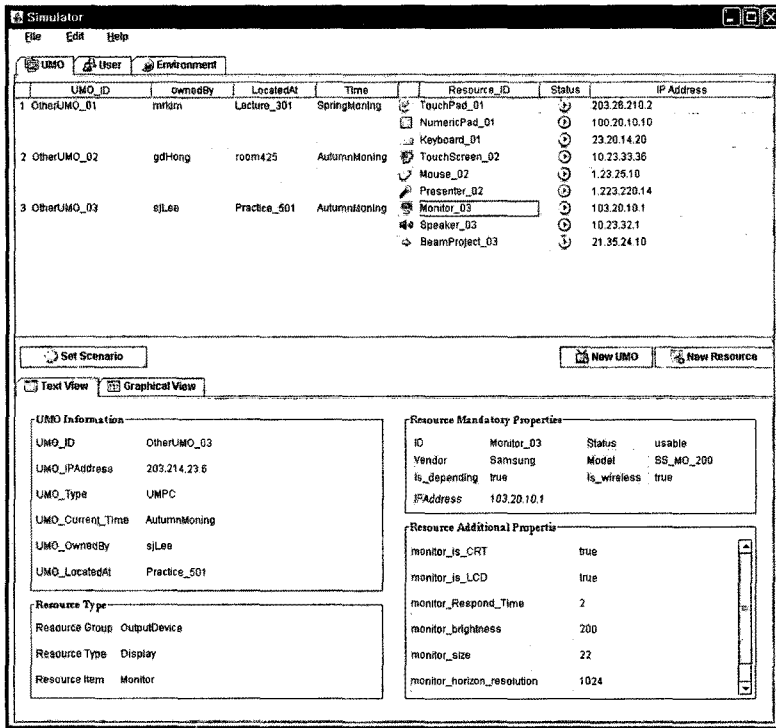


그림 8 UMO, 자원 정보 출력

• 상황 정보 온톨로지의 자동 생성

그림 8은 외부 USim의 메인 윈도우 화면이다. 윈도우는 크게 3개의 탭으로 구성된다. UMO 탭에서 상단 테이블에서는 USim에서 생성한 주변 UMO와 각 UMO가 가지는 자원 목록을 나열한다. 중간에는 시나리오 설정, 새로운 UMO 추가, 새로운 리소스 추가 버튼이 있다. 마지막으로 하단에는 상단 목록에서 선택된 자원과 UMO의 상세 정보는 윈도우의 하단의 TextView와 Graphical View에 모두 출력된다.

그림 9와 같이 New UMO 버튼을 누르면, 새로운 UMO의 추가를 위한 정보 입력 다이얼로그가 출력된다. UMO를 추가하거나 수정할 때, OwnedBy, CurrentTime, LocatedAt는 현재 존재하는 사용자, 시간, 위치 온톨로지 인스턴스의 Subject를 제공하여 쉽게 관계를 맺을 수 있게 한다. 윈도우 하단의 UMO Information의 선택하면 UMO 정보 수정 다이얼로그가 출력되어 정보를 수정할 수 있다.

New Resource 버튼을 누르면 그림 10과 같이 새롭게 추가할 자원의 종류를 선택하는 다이얼로그가 제공되고, 선택된 자원의 종류에 맞는 정보 입력 다이얼로그가 출력된다. Resource Mandatory Properties는 자원이 필수적으로 가져야 하는 속성을 의미하고, Resource Additional Properties는 해당 자원이 고유하게 보유하

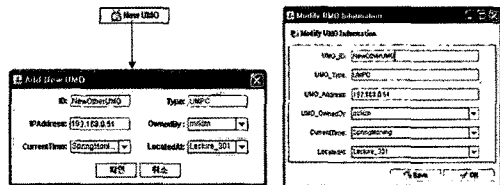


그림 9 UMO 추가, 수정 다이얼로그

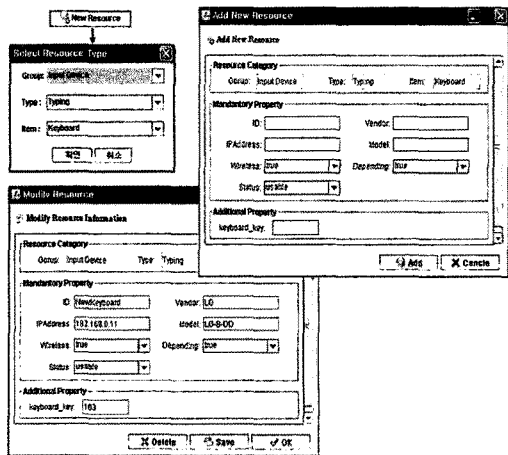


그림 10 자원 추가, 수정, 삭제 다이얼로그

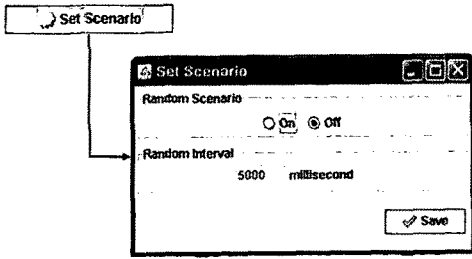


그림 11 자동 시나리오 구성 설정

는 속성 정보를 나타낸다. UMO 탭의 윈도우 하단에서 Resource Mandatory Properties 또는 Resource Additional Properties를 선택하면 자원 정보 수정 다이얼로그가 출력되어 정보를 수정 및 삭제할 수 있다.

그림 11과 같이 UMO 탭에서 Set Scenario 버튼을 선택한 경우 출력되는 시나리오 설정 다이얼로그이다. 시나리오 설정은 이전에 설명한 자원을 하나씩 추가하고 수정하는 방법과 다르게 설정한 주기에 따라 지속적으로 다양한 상황 정보를 자동으로 생성해준다. 이를 통해 예상하지 않았던 다양한 상황 정보들을 생성할 수 있고, 신속하게 상황 정보를 자동으로 변경할 수 있다.

그림 12는 User 탭을 선택한 경우 나타나는 화면으로, 왼쪽 User Profile 테이블에는 생성한 사용자 프로파일

목록을 나타낸다. 오른쪽 User Preference 테이블에는 사용자의 디바이스에 대한 선호도 목록을 출력한다.

NewUser 버튼을 누르면 그림 13과 같이 새로운 사용자 정보를 입력할 수 있는 다이얼로그를 출력한다. UMO 탭과 마찬가지로, 정보의 수정 및 삭제는 테이블에서 해당 목록을 선택하여, 수정 및 삭제 다이얼로그에서 수행한다.

New Preference 버튼을 누르면, 그림 14와 같이 자원에 대한 사용자의 선호도 정보를 입력 할 수 있는 다이얼로그를 출력한다. 선호도 정보의 수정 및 삭제는 User Profile 테이블에서 수정 및 삭제할 목록의 선택을 통해 가능하다.

그림 15는 Environment 탭의 화면 구성 모습이다. 상단의 Location Context는 가상으로 생성한 장소에 대한 정보를 목록을 출력한다. 그리고 Time Context에서는 가상으로 생성한 시간 정보를 목록으로 출력한다.

새로운 위치정보는 New Location 버튼을 누르면, 그림 16과 같이 정보 입력 다이얼로그가 출력되어 새로운 위치 정보 온톨로지를 생성할 수 있다. 위치 정보의 수정 및 삭제는 해당 목록을 선택하면, 수정 및 삭제 다이얼로그를 통해 수행할 수 있다.

새로운 시간 정보는 New Time 버튼을 누르면, 그림 17과 같이 정보 입력 다이얼로그가 출력되어 새로운 환경 시간 정보 온톨로지를 추가할 수 있다. 시간 정보의

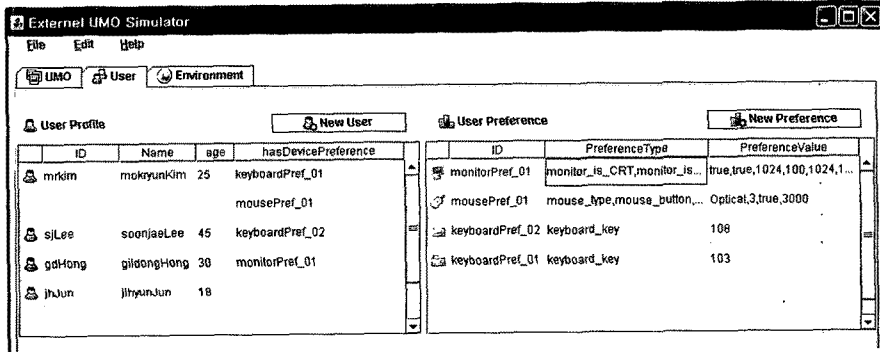


그림 12 사용자 정보 출력

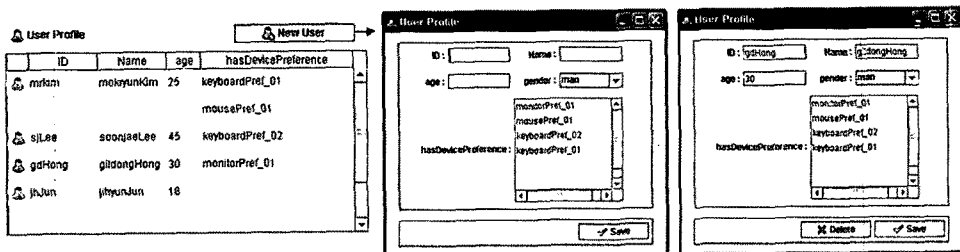


그림 13 사용자 프로파일 추가, 수정, 삭제 다이얼로그

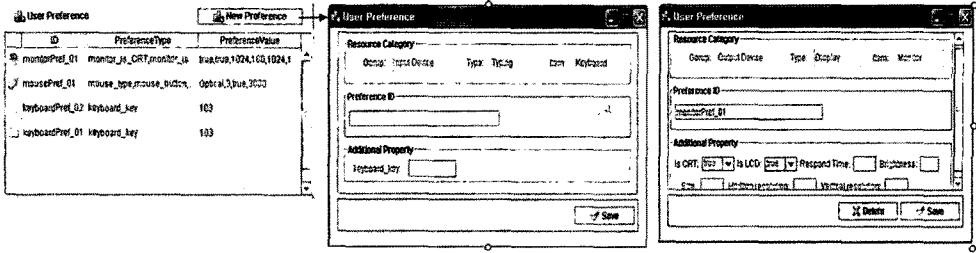


그림 14 사용자 자원 선호도 추가, 수정, 삭제 다이얼로그

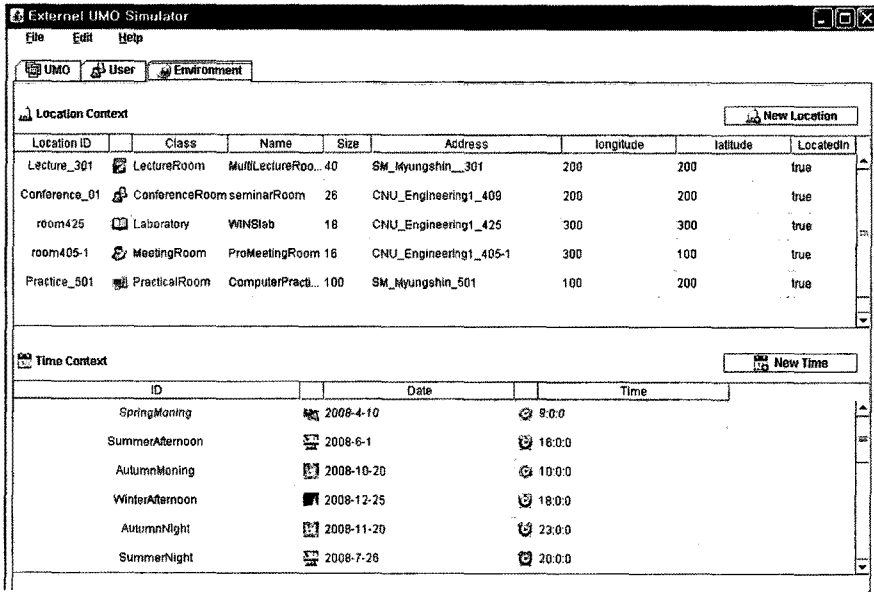


그림 15 위치, 시간 정보 출력

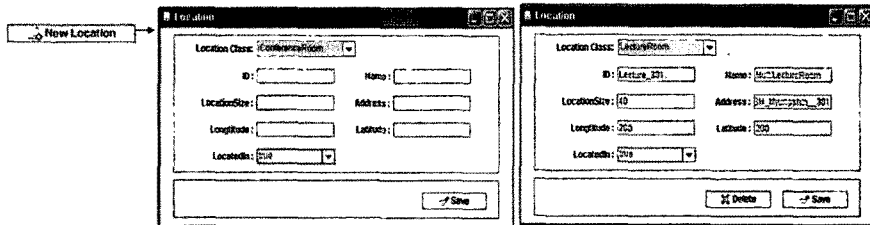


그림 16 위치 추가, 수정, 삭제 다이얼로그

수정 및 삭제는 해당 목록을 선택하면, 수정 및 삭제 다이얼로그를 통해 수행할 수 있다.

내부 USim은 외부 USim의 User, Environment 탭을 하나로 통합하고, 추가로 Query 탭이 존재한다. 그림 18은 쿼리 탭의 인터페이스 화면이다.

그림 18의 상단에 있는 서비스 요청 버튼을 누르면, 현재 요청 가능한 서비스인 프레젠테이션, 문서편집, 동영상재생을 선택할 수 있다. 이는 추후 온톨로지 확장을

통해 다양한 서비스를 추가할 수 있다. 그림 18은 동영상재생 서비스의 요청 결과를 나타낸다. 디바이스 검색 결과는 주변 UMO로부터 동영상재생에 필요한 자원인 모니터를 검색한 결과이다. 컨텍스트 기반 1차 디바이스 추론 결과는 외부 UMO 디바이스 검색 결과를 바탕으로 내부 UMO의 위치 및 시간을 고려하여 UMO와 가깝고 현재 설정된 시간에 알맞은 디바이스를 추론한 결과를 나타낸다. 사용자 정보 기반 2차 디바이스 추론 결

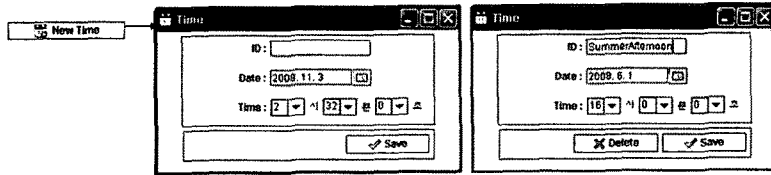


그림 17 시간 추가, 수정, 삭제 다이얼로그

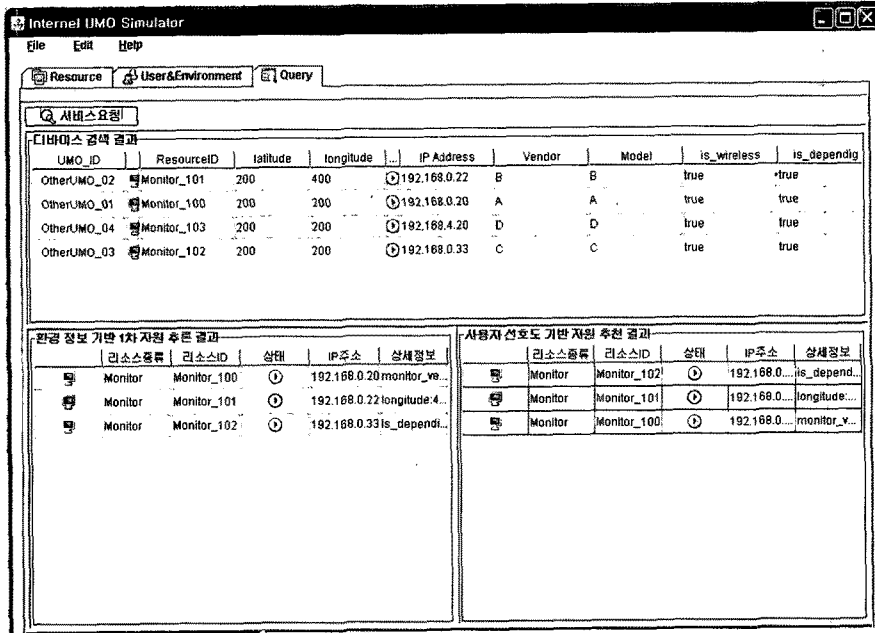


그림 18 자원 검색 및 추론 결과

과는 1차 디바이스 추론 결과를 바탕으로 UMO 사용자의 모니터 선호도에 부합하는 모니터 순으로 정렬한 결과를 나타낸다. 실제 UMO에서 선호도에 기반하여 정렬한 모니터 목록만이 사용자에게 추천되며, 사용자는 추천된 모니터 중 이용하고자 하는 모니터를 선택한다.

4. USim 검증

본 절에서는 USim의 실험 시나리오를 통해 실제 UMO에서 추론 및 추천되는 자원과 USim을 통해 구성된 상황 정보에서 추론 및 추천된 자원의 일치 여부를 확인한다. 이를 통해 실제 UMO 대신하여 USim을 통해 다양한 환경에서 자원 추론 및 선호도에 맞는 자원 추천이 가능함을 입증 할 수 있다.

먼저 프레젠테이션을 수행하기 위해 수집된 자원들 중에서 현재 환경 정보에 적합한 자원을 추론하기 위한 추론 규칙은 총 34개가 존재한다. 그 중 핵심적인 추론 규칙은 아래와 같다. 아래 추론규칙 중에서 추론규칙 5

는 온톨로지를 사용하기 때문에 가능한 추론규칙이라 할 수 있다. 온톨로지는 계층적인 구조를 갖기 때문에, 추론 규칙 5와 같이 프리젠테이션에 필요한 포인팅 자원인 프리젠테이션, 마우스, 키보드가 없는 경우, 그 상위 개념인 포인팅 클래스에 있는 다른 자원을 선택하도록 한다. 아래와 같은 다양한 추론 규칙에 따라 적합한 자원이 자동으로 추론 된다.

추론 규칙 1. UMO와의 거리가 1 이하인 프리젠테이션이 있다면 선택

```
(defrule PresentationPresenter
  (Presenter (id ?pID) {distanceLevel <= 1})
  => (assert (Presentation ?pID))
)
```

추론 규칙 2. 추론 규칙 1에 만족하는 프리젠테이션이 없는 경우, 거리를 확장하여 선택

```
(defrule PresentationPresenterfaraway
  (not(Presenter (id ?pID) {distanceLevel <= 3}))
  (Presenter (id ?pID) {distanceLevel <= 5 &&
```

```

distanceLevel > 3})
=> (assert (Presentation ?pID))
)
    
```

추론 규칙 3. 프리젠티가 없는 경우, 마우스를 선택

```

(defrule PresentationMouse
(not(Presenter (id ?id)))
(Mouse (id ?mouID){distanceLevel <= 1})
=>(assert (Presentation ?mouID))
)
    
```

추론 규칙 4. 프리젠티, 마우스, 키보드 모두 없는 경우, 터치패드 선택

```

(defrule PresentationTouchpad
(and (not(Presenter (id ?id)))
(not(Mouse (id ?mid)))
(not(Keyboard (id ?mid))))
(Touchpad (id ?tID) {distanceLevel <= 1})
=>(assert (Presentation ?tID))
)
    
```

추론 규칙 5. 장소의 크기가 5이상인 경우, 마이크를 선택

```

(defrule PresentationMicrophone
(ConferenceRoom {locationSize >= 5})
(Microphone (id ?micID) {distanceLevel <= 1})
=>(assert (Presentation ?micID))
)
    
```

추론 규칙 6. 프리젠티, 마우스, 키보드 모두 없는 경우, 포인팅 클래스에 속하는 자원을 선택

```

(defrule PresentationTouchpad
(and (not(Presenter (id ?id)))
(not(Mouse (id ?mid)))
(not(Keyboard (id ?mid))))
(Pointing (id ?tID) {distanceLevel <= 1})
=>(assert (Presentation ?tID))
)
    
```

추론 규칙 7. 장소의 크기가 5이상인 경우, 빔 프로젝터를 선택

```

(defrule PresentationBeamproject
(ConferenceRoom {locationSize >= 5})
(BeamProject (id ?bID) {distanceLevel <= 1})
=>(assert (Presentation ?bID))
)
    
```

추론 규칙 8. 빔 프로젝터가 없거나, 장소의 크기가 5 이하이고, 현재 시간이 낮이라면 밝기가 높은 모니터를 선택

```

(defrule PresentationMonitorForDay
(or (not(BeamProject (id ?id)))
(ConferenceRoom {locationSize < 5}))
(Time {currentTime <= 18 && currentTime >= 6})
(Monitor {monitor_brightness >= 300} {distanceLevel <= 1}{id ?mID})
=>(assert (Presentation ?mID))
)
    
```

UMO1	보유 자원		자원속성	
프리젠티A	버튼 수(개)			
	3			
마이크A	타입	Sensitivity(dB)		
	Handheld	-75		
모니터A	해상도	밝기	크기	
	800*600	300	17	
마우스A	버튼 수(개)	타입	휠	
	2	optical	true	

UMO2	보유 자원	속성		
빔프로젝터B	is_fixed			
	true			
모니터B	해상도	밝기	크기	
	1024*768	500	24	
마우스B	버튼 수(개)	타입	휠	
	2	ball	true	

UMO3	보유 자원	속성		
마이크C	타입	Sensitivity(dB)		
	Handheld	-48		
모니터C	해상도	밝기	크기	
	1248*1024	500	24	
터치패드C				

UMO4	보유 자원	속성		
모니터D	해상도	밝기	크기	
	1024*768	200	24	

그림 19 주변 UMO의 보유 자원 정보

시간	장소(크기)	위치	
		위도	경도
10:00(오전)	10평	200	200

그림 20 로컬 UMO의 환경 정보

실험 환경은 다음과 같다. 먼저 주변에 총 4개의 UMO가 존재하고, 각 UMO가 가지는 자원에 대한 정보는 그림 19와 같다. 현재 프리젠티 1개, 마이크 2개, 모니터 4개, 마우스 2개, 터치패드 1개가 존재한다. 자원 종류는 동일하지만, 그림 19와 같이 각 자원이 가지는 속성은 차이가 있다.

로컬 UMO의 초기 환경 정보는 그림 20과 같다. 현재 시간은 오전 10시이며, UMO가 있는 장소의 크기는 10평, 장소내의 UMO 위치는 위도 200, 경도 300이다.

로컬 UMO 소유자의 자원 선호도 정보는 그림 21과 같다. 마우스는 Optical형을 선호하고, 모니터는 1248*1023, 밝기는 500, 크기는 24인치인 모니터를 선호한다. 그리고 마이크는 48 dB의 Sensitivity를 갖는 마이크를 선호한다.

그림 22는 실제적인 테스트 환경을 구축하고 시나리오의 변경에 따라 추론 및 추천되는 자원의 결과이다. 추론 규칙에 맞게 자원이 추론 되고, 사용자 선호도에

마우스			모니터			마이크
머튼 수(개)	타입	컬	해상도	밝기	크기(mch)	Sensitivity(dB)
2	Optical	True	1248*1024	500	24	48

그림 21 사용자의 자원 선호도 정보

시나리오	변경 사항	추론 자원	자원 추천순위
시나리오1	-	프리젠티A, 빔프로젝터B, 마이크A, 마이크C	프리젠티A, 빔프로젝터B, 마이크C, 마이크A
시나리오2	프리젠티A 공유 불가	마우스A, 마우스B, 빔프로젝터B, 마이크A, 마이크C	마우스B, 마우스A, 빔프로젝터B, 마이크C, 마이크A
시나리오3	마우스 A,B 공유 불가	터치패드C, 빔프로젝터B, 마이크A, 마이크C	터치패드C, 빔프로젝터B, 마이크C, 마이크A
시나리오4	장소 크기 변경	모니터 A, B, C, 터치패드C	모니터 C, B, A, 터치패드C
시나리오5	사용자의 모니터 해상도 선호도 1024*768로 변경	모니터 A, B, C, 터치패드C	모니터 B, C, A, 터치패드C
시나리오6	시간 변경	모니터D, 터치패드C	모니터D, 터치패드C

그림 22 시나리오에 따른 UMO의 자원 추론 및 추천 결과

시나리오	변경 사항	추론 자원	자원 추천순위	일치율
시나리오1	-	프리젠티A, 빔프로젝터B, 마이크A, 마이크C	프리젠티A, 빔프로젝터B, 마이크C, 마이크A	100%
시나리오2	프리젠티A 공유 불가	마우스A, 마우스B, 빔프로젝터B, 마이크A, 마이크C	마우스B, 마우스A, 빔프로젝터B, 마이크C, 마이크A	100%
시나리오3	마우스 A,B 공유 불가	터치패드C, 빔프로젝터B, 마이크A, 마이크C	터치패드C, 빔프로젝터B, 마이크C, 마이크A	100%
시나리오4	장소 크기 변경	모니터 A, B, C, 터치패드C	모니터 C, B, A, 터치패드C	100%
시나리오5	사용자의 모니터 해상도 선호도 1024*768로 변경	모니터 A, B, C, 터치패드C	모니터 B, C, A, 터치패드C	100%
시나리오6	시간 변경	모니터D, 터치패드C	모니터D, 터치패드C	100%

그림 23 시나리오에 따른 USim의 자원 추론 및 추천 결과

맞게 자원 추천 순위가 결정됨을 확인 할 수 있다.

그림 23은 실제 UMO를 대신하여 USim에서 시나리오를 변경하며 추론 및 추천되는 자원의 결과를 나타낸다. 그림 23에서 알 수 있듯이, USim에서 추천된 자원이 실제 UMO에서 추천하는 자원과 일치함을 확인할 수 있다.

실험을 통해 알 수 있듯이, 실제 UMO와 동일한 추론 및 추천 결과를 나타낸다. 따라서 상황 정보를 구성하기 어려운 실제 환경에서 추론을 검증하는 것 보다 다양한 상황 정보를 생성하고 조작할 수 있는 USim을 이용하는 것이 더 효과적임을 알 수 있다.

5. 결론

본 논문에서는 모바일 단말기의 제한적인 자원의 문제를 해결하기 위하여 주변의 자원을 실시간으로 공유하는 연구인 UMO 프로젝트에서, UMO의 신뢰성 있는

추론의 동작 검증을 위해 가상의 다양한 자원, 사용자 프로파일, 선호도, 환경 온톨로지 정보를 생성할 수 있는 추론 검증 도구인 USim을 설계 및 개발하였다. USim은 내부 USim과 외부 USim으로 구성하여 내부 USim은 보유 자원 및 환경, UMO 사용자의 프로파일, 선호도를 다양하게 구성하고, 서비스의 요청 및 자원 추론, 추천 결과를 제공하였다. 외부 USim은 다수의 주변 UMO 정보를 생성하고 UMO의 자원 및 환경 정보를 다양하게 생성, 수정, 삭제할 수 있도록 제공하였다. 그리고 USim은 그래픽 사용자 인터페이스를 제공하여 가상의 상황 및 자원 정보를 편리하게 생성할 수 있는 사용 환경을 제공하였다. USim은 실제 UMO에서 실험하기 어려운 다양한 상황 정보의 생성하고, 추가 및 수정된 추론 규칙을 실제 UMO의 추론 엔진에 적용하기 전에 USim을 통해 검증해 봄으로써, 실제 UMO를 이용

한 자원 추천 테스트의 시간 및 비용을 줄일 수 있다. 마지막으로 실험을 통해 USim에서 추천된 자원이 실제 UMO에서 추천하는 자원과 일치함 확인하여 USim에서 생성해 내는 다양한 상황 정보가 실제 UMO에서 생성되는 상황 정보와 실제로 일치함을 확인하였다.

참 고 문 헌

[1] C. Taylor and L. Dajani, "The future of homecare systems in the context of the ubiquitous web and its related mobile technologies," In *Proc. the 1st ACM international conference on Pervasive Technologies Related to Assistive Environments*, vol. 282, pp.1-4, 2008.

[2] MC Lee, HK Jang, YS Paik, SE Jin and S Lee, "Ubiquitous Device Collaboration Framework in Ubiquitous Environment: Celadon," In *Proc. the fourth IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems and the Second International Workshop on Collaborative Computing, Integration, and Assurance (SEUS-WCCIA'06)*, pp.141-146, 2006.

[3] H. Chen, T. Finin, and A. Joshi, "An Ontology for Context-Aware Pervasive Computing Environments," In *Proc. Cambridge University Press*, vol.18, no.3, pp.197-207, 2004.

[4] A. Ranganathan and Roy H. Campbell, "A Middleware for Context Aware Agents in Ubiquitous Computing Environments," In *Proc. Middleware 2003*, vol.2672, p.998, 2003.

[5] Jong-Hyun Park, Sun-Hee Kang and Ji-Hoon Kang, "Personalized Resource Recommender System Based on Context-Aware in Ubiquitous Environments," In *Proc. The Journal of Korea Computer Congress(KCC) 2009*, vol.32, no.1(C), pp.95-99, 2008.

[6] Won-Ik Park, Jong-Hyun Park, Young-Kuk Kim and Ji-Hoon Kang, "A Context-Based Device Collaboration System in Ubiquitous Environments," In *Proc. The Journal of Electronics Engineers of Korea*, vol.3, no.11, pp.86-96, 2008.



박 영 호

1986년 3월~1992년 2월 동국대학교 공과대학 컴퓨터공학과(학사, 공학석사). 1993년 8월~1999년 2월 한국전자통신연구원(ETRI) 교환전송연구단 선임연구원. 1999년 3월~2005년 8월 한국과학기술원(KAIST) 전산학과(공학박사, DBMS). 2005년 9월~2006년 2월 한국과학기술원(KAIST) 첨단정보기술연구센터 연구원. 2001년 9월~2006년 2월 한국산업기술대학교 컴퓨터공학과 겸임교수. 2005년 9월~2006년 2월 동국대학교 컴퓨터멀티미디어학과 겸임교수. 2006년 3월~현재 숙명여자대학교 이과대학 멀티미디어학과와 조교수. 관심분야는 데이터베이스, XML, IR(정보검색), 멀티미디어데이터베이스, Bio정보공학, 영상미디어, 예술&공학인터페이스



김 목 련

2003년~2007년 숙명여자대학교 멀티미디어학과(학사). 2007년~2009년 숙명여자대학교 멀티미디어학과(석사). 관심분야는 내용기반 이미지 검색, XML 검색