유비쿼터스 환경에서의 상황 기반 디바이스 추천 시스템[1]

최환수[0], 강선희, 이용대, 장서윤, 박원익, 박종현, 김영국, 강지훈
충남대학교 컴퓨터공학과
e-mail : {h_s_choi[0], shkang, racunda, pineai, wonik78, jonghyunpark, ykim, jhkang}@cnu.ac.kr

A Context-based Device Recommendation System in Ubiquitous Environments

Hwan-Soo Choi[O], Sun-Hee Kang, Yong-Dae Lee, Seo-Yoon Jang, Won-Ik Park, Jong-Hyun Park, Young-Kuk Kim, Ji-Hoon Kang,
Dept. of Computer Engineering, ChungNam National University

요 약

유비쿼터스 컴퓨팅 환경이란 생활 속에 존재하는 컴퓨팅 자원들을 이용해 사용자가 언제 어디에서든 편리한 서비스를 제공받을 수 있는 환경을 의미한다. 유비쿼터스 환경에 존재하는 수없이 많고 다양한 컴퓨팅 자원들을 사용자가 최적으로 사용하기 위해서는 사용자가 어떤 상황에 있으며 이때 어떤 자원이 사용자의 현 상황에서 가장 적절한지를 판단하는 것이 반드시 필요하다.

본 논문에서는 이를 위하여 사용자가 현재 존재하는 유비쿼터스 공간에서 사용자의 상황을 인식하고 사용 가능한 최적의 자원들을 공유할 수 있도록 추천해주는 상황 기반 디바이스 추천 시스템을 제안한다. 우리의 추천 시스템은 상황에 따른 사용자 개개인의 특성이 고려된 사용자의 개인 정보 및 규칙들을 이용하여 사용자의 상황에 최적의 디바이스를 추론한다. 향후 제안한 서비스 추론 방법은 유비쿼터스 환경에서 더 나은 개인화 서비스 시스템의 개발 및 운용을 효과적으로 지원할 수 있을 것으로 기대된다.

1. 서론

최근 “유비쿼터스 컴퓨팅”이라는 지능형 서비스 프레임 워크가 제안되고 있다. 유비쿼터스 컴퓨팅의 의미는 인간의 실제 세계에 산재해 있는 컴퓨팅 장치들과 인간을 자연스럽게 상호작용하도록 하는 것이다[1]. 이러한 유비쿼터스 컴퓨팅 환경은 다음과 같다. 첫째, 모든 컴퓨터는 네트워크를 통해 서로 연결되어야 하며, 둘째, 이용자의 눈에 보이지 않아야 하며, 셋째, 언제 어디서나 사용 가능해야 하며, 마지막으로, 현실세계의 모든 사물과 환경 속으로 스며들어 일상생활과 통합되어야 한다 [2].

이러한 유비쿼터스 환경을 USS(Ubiquitous Smart Space)라 한다. 인간은 USS 환경을 언제 어디서나 제공받을 수 있게 하기 위해 모바일 디바이스가 필요하다. 하지만 모바일 디바이스는 제한적인 자원으로 인해 USS 환경의 모든 자원을 가지고 있을 수가 없다. 따라서, 본 논문에서는 상황인지를 통해서 사용자가 요구하는 서비스에 따른 필요한 디바이스 정보를 모바일 디바이스에 알려주는 유비쿼터스 환경에서 상황 기반 디바이스 추천 시스템을 제안하고 있다.

제안한 시스템은 각 개인이 모바일 디바이스를 가지고 있다는 장점을 이용하여 사용자 개개인에게 현재 상황에서의 최적의 디바이스를 찾아줄 수 있도록 도와준다.

본 논문의 구성은 다음과 같다. 2 절에서는 디바이스 추천 시스템에 대한 기존의 연구를 기술하고 본 논문의 시스템과 차이점에 대해 설명하고, 3 절에서는 상황 기반 디바이스 추천 시스템의 구조 및 구현에 대하여 설명하고 4 절에서는 우리의 시스템에 맞춰 시나리오를 제시하고 디바이스 추천을 위한 규칙을 보인다. 마지막으로 5 절에서는 결론을 기술한다

2. 관련 연구

유비쿼터스 컴퓨팅에 대한 관심이 증가하면서 유비쿼터스 컴퓨팅 환경에서 사용자는 자신에게 맞는 디바이스를 통해 서비스를 받기를 원한다[3]. 이를 위해 디바이스들간의 협력은 필수적이다. UPnP, JINI는 이러한 디바이스들간의 통신을 위해 제안된 대표적인 표준 인터페이스로서 네트워크 환경에서 ‘플래그 앤 플레이’가 가능하도록 하는 기술이다[4,5].

UPnP[10]는 홈 네트워크 환경에서 운영체제, 언어 및 하드웨어에 독립적인 서비스 환경을 제공하는 미들웨어로 단순하고 유연하며 표준에 기반한 peer-to-peer 방식의 연결성을 제공하여 사용자는 단지 장치를 네트워크에 연결시켜 주면 네트워크 상에 연결된 기존의 장치들이 자동으로 새로 추가된 장치를 발견

본 연구는 21 세기 프론티어 연구개발사업의 일환으로 추진되고 있는 정보통신부의 유비쿼터스컴퓨팅및네트워크원천기반기술개발사업의 지원에 의한 것임

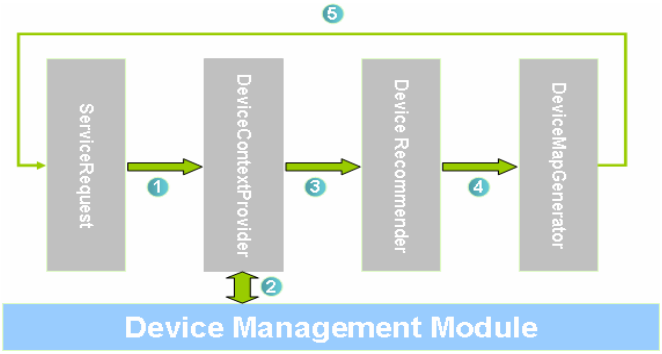
하여 제어하거나 다른 장치가 가진 서비스를 찾을 수 있다[6].

JINI[11]는 SUN 에서 개발한 분산 환경의 홈 네트워크 에서 자원 공유를 제공하는 미들웨어 기술로 네트워크 상의 모든 종류의 디바이스와 소프트웨어 자원의 통합체를 구성하여 서비스와 자원을 공유하고 사용자의 위치에 관계없이 용이한 자원의 접근 및 네트워크의 개설, 갱신, 변경 작업의 단순화를 목표로 한다[6].

제안된 시스템은 주변의 디바이스를 활용할 수 있다는 면에서 UPnP, JINI 와 유사할 수 있지만 상황에 따라 사용자에 맞는 디바이스를 자동으로 추천해준다는 점에서 다르다고 말할 수 있다.

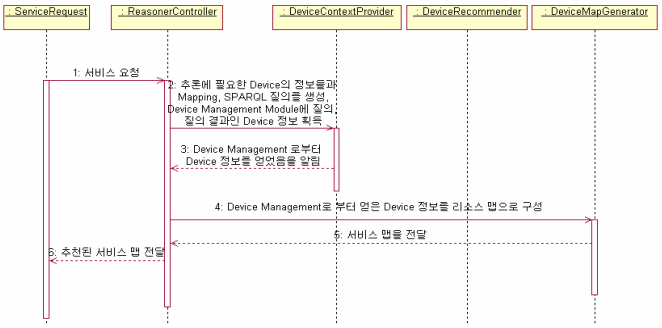
3. 상황 기반 디바이스 추천 시스템 설계 및 구현

상황 기반 디바이스 추천 시스템의 역할은 사용자가 요청한 서비스를 제공하기 위해 사용자의 상황에 따라 필요한 디바이스를 전문가 시스템을 사용해서 효율적으로 찾는 것이다. (그림 1)은 상황 기반 추천 시스템의 구조이다.



(그림 1) 상황 기반 추천 시스템의 구조

(그림 1)에서처럼 본 연구에서 제안하고 있는 상황 기반 디바이스 추천 시스템은 다양한 USS 에 적합하도록 각 모듈의 확장을 고려해서 설계하였다.

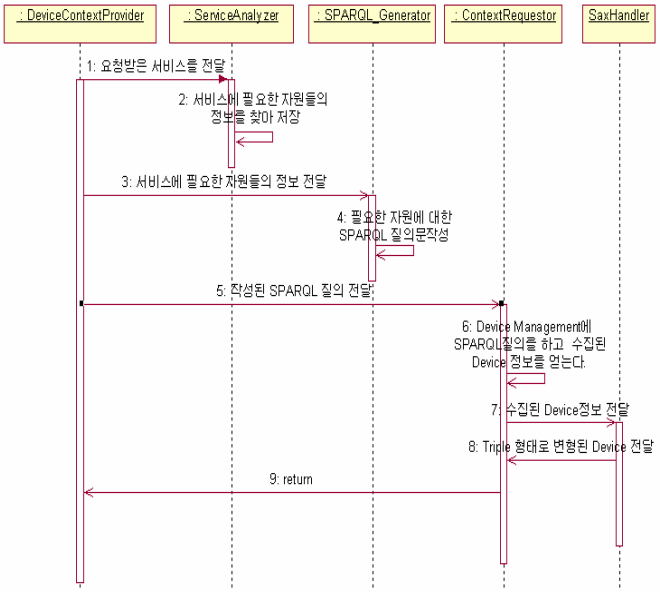


(그림 2) 상황기반추천시스템 시퀀스 다이어그램

(그림 2)는 상황 기반의 추천 시스템에 대한 전체적인 흐름을 보여주고 있다. ServiceRequest 에서 사용자로부터 서비스 요청이 들어오면 그 정보를 DeviceContextProvider 로 전달한다. Device Context Provider 는 전달받은 서비스에 대한 필요한 디바이스 정보를 얻고 Device Management Module 을 통해 사용 가능한 디바이스 정보를 얻는다. 얻어진 사용 가능한 디바이스 정보들은 Device Recommender 로 전달되고 Device Recommender 에서는 사용자 정보를 이용하여 사용 가능한 디바이스 중 사용자에 맞는 디바이스를 가려낸다. DeviceMap Generator 에서는 추천된 디바이스를 동작하는데 있어 필요한 정보만을 뽑아 Map 으로 생성하여 전달한다.

3.1 DeviceContextProvider

DeviceContextProvider 는 ServiceRequest 로부터 입력 된 서비스 요청을 처리하기 위해 동작하는 모듈로써 크게 4 개의 서브 모듈로 나뉜다. (그림 3) Device ContextProvider 의 순차적인 흐름도이다.

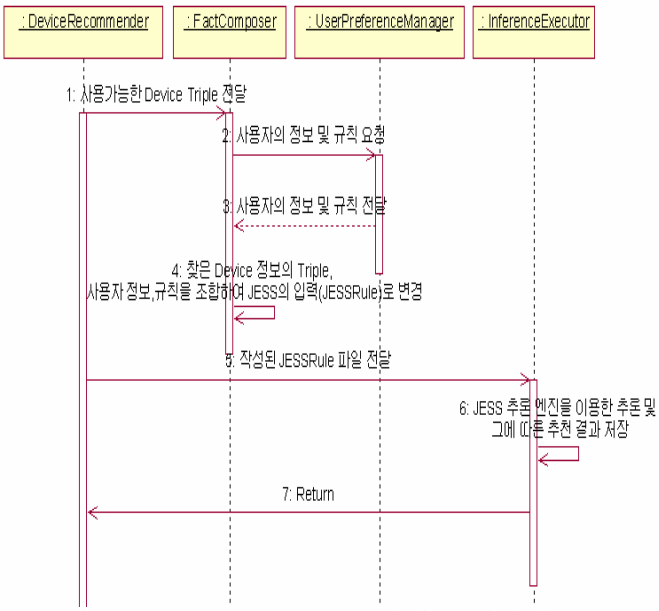


(그림 3) DeviceContextProvider 시퀀스 다이어그램

궁극적으로 서비스에 필요한 디바이스를 Device Management Module 에 요청하기 위한 질의를 생성하고 질의의 결과인 사용 가능한 디바이스 정보를 얻는 일을 담당한다. 이를 위해서 사용자의 서비스 요청에 따른 필요한 디바이스를 찾는 Service Analyzer 모듈과 Device Management Module 에 필요한 디바이스를 요청하기 위한 질의 생성기인 SPARQL[9]_Generator 모듈, 그리고 Device Management Module 에 SPARQL 로 질의하고 그 결과를 취하는 모듈인 Context Requestor 모듈로 구성된다.

3.2 DeviceRecommender

DeviceRecommender 는 Device Management Module 로부터 얻은 이용 가능한 자원 정보들 중 사용자 요구에 맞는 디바이스를 추천 해주는 모듈로 크게 3 개의 서브 모듈로 나뉜다. [그림 4] Device Recommender 의 순차적인 흐름도이다.



(그림 4) DeviceRecommender 시퀀스 다이어그램

디바이스를 추천하기 위해 우리의 시스템은 자바 환경에서 규칙(Rule) 기반 추론 엔진인 JESS[7,8]를 이용했다. JESS 추론 엔진의 입력은 JESS Function 혹은 JESS ML 로 기술된 규칙이다. 그러므로 추론을 위해서는 JESS 추론 엔진에서 요청하는 형태의 규칙을 생성 해야한다. FactComposer 모듈은 Device Management Module 로부터 얻은 디바이스 정보의 triple 구조와 사용자 선호도 정보와 규칙, 세가지 정보들을 취합하여 JESS 추론 엔진의 입력 형태로 변형한다. User PreferenceManager 모듈은 사용자의 정보를 저장하고 이를 제공하는 역할을 한다. 여기에는 사용자의 신상정보, 선호도에 대한 정보와 사용자가 정의한 규칙이 저장되어 있다. 이 정보들은 사용자 에게 개인화된 서비스를 제공하기 위해서 사용된다. 그리고, Inference Executor 모듈은 작성된 JESS 입력 값을 JESS 추론 엔진에 질의하여 추론 결과를 얻는 역할을 담당한다.

3.3 DeviceMapGenerator

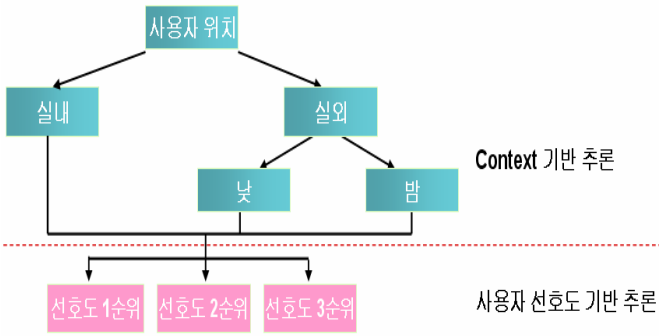
DeviceMapGenerator 는 JESS 추론엔진으로부터 추론된 디바이스를 동작하는데 있어 필요한 정보만을 추출해내기 위한 모듈로 추천된 디바이스의 종류, IP, ID 정보들을 추출하여 Map 으로 구성하는 역할을 한다.

4. 실험 시나리오

본 장에서는 예제 시나리오를 이용하여 본 논문에서 제안한 상황 기반의 디바이스 추천 시스템에 적용시켜본다.

Kim 이라는 사람이 밤 시간이 되어 사무실에 나와서 모바일 디바이스를 통해 동영상을 보면서 집으로 가기 위해 버스정류장으로 가고 있다. 버스정류장에는 USS 환경의 모니터들이 설치되어 있다. 버스정류장에 도착한 Kim 은 좀 더 나은 화면으로 동영상을 보기 위해 디스플레이 관련 디바이스를 요청하게 되고 현재 상황에 맞게 Kim 이 원하는 모니터를 찾아내 그 모니터로 동영상을 감상할 수 있도록 도와준다.

본 시나리오에서 상황 기반의 디바이스 추천 시스템은 다음과 같이 진행된다. Service Request 모듈에서 Display 를 요청하면 DeviceContextProvider Module 에서는 Monitor 가 필요하다는 것은 인지하고 이를 Device Management Module 에 요청을 하여 버스정류장 의 Monitor 들에 대한 정보를 얻어온다. 그 정보는 Device Recommender 모듈에서 사용자에게 적합한 모니터를 찾아내어 DeviceMapGenerator 를 통해 Kim 의 모바일과 모니터 연결에 필요한 정보만을 추출하여 그 정보를 전달한다. (그림 5)는 시나리오를 위한 규칙 기반 추론 구조이다.



(그림 5) 사용자 규칙 기반 추론 구조

사용자 선호도 정보에 맞게 서비스를 해주기 위해 본 논문은 현재 주변 상황정보와 사용자 선호도 정보를 모두 고려한다. 상황에 따른 사용자의 선호도를 파악하고 이에 맞는 서비스를 추천해준다. 만약 결과가 나오지 않는다면 상황에 맞는 서비스를 추천을 해주게 된다.

4.1 상황 기반 추론

상황 기반 추론은 사용자에게 자원을 추천할 때, 가장 먼저 고려해야 하는 추론이다. 본 논문에서는 사용자의 현재 위치와 현재 시간에 따라 자원에 대한 사용자의 선호도가 변경된다는 가정을 했다. [표 1]은 사용자의 현재 위치 건물 밖이고 밤이라는 조건에서 모니터 선택 기준의 우선순위를 찾는 규칙이다.

```
(defrule FindPreferencePriorityOrder
 (FollowedInPlaceUserPreferenceRank
  (LocationAt In) (Day_type  Night) (First ?first)
   (Second ?second)(Third ?third) )
 =>
 (?*PreferenceRank* add ?first)
 (?*PreferenceRank* add ?second)
 (?*PreferenceRank* add ?third)
 (store Rank ?*PreferenceRank*))
```

[표 1] 시간과 장소에 따른 사용자의 선호도를 찾는 규칙

[표 1]에서 규칙을 이용하여 현재 상황에 맞는 선호도 정보를 찾게 되면 그 정보에 관련된 사용자 선호도 값을 적용 시킬 필요가 있다.

4.2 사용자 선호도 기반 추론

우리는 상황 기반 추론을 통해 특정 자원에 대한 속성의 우선순위를 얻을 수 있었다. 사용자 선호도 기반 추론은 우선순위가 높은 속성에 대한 사용자 선호도 정보를 이용한 추론이다. [표 2]은 사용자의 선호도가 모니터의 밝기일 때 현재 시간을 고려한 규칙이다.

```
(defrule OutSideNightPreferredMonitorsByBrightness
 (UserLocation Out)
 (Monitor (location Out) (ID ?id)
  (Monitor_ brightness ?brightness) )
 (Time (Day_type Night))
 (UserPrefereneceForLocation
  (PreferenceOrder
   Monitor_Brightness ?second ?third) )
 (User_Preference
  (PreferredMonitor_Brightness ?preferredBrightness) )
 (test (>= ?brightness ?preferredBrightness) )
 (test (<= ?brightness (+  ?preferredBrightness 100) ) )
   =>
 (assert (UserBasedRecommendMonitor ?id)))
```

[표 2] 사용자의 선호도가 모니터의 밝기일 때 현재 시간을 고려한 규칙

[표 2]를 통해 상황에 따른 사용자 선호도를 고려한 모니터가 추천되게 된다. 추천된 모니터의 정보 중에서 실제 서비스에 필요한 요소만을 뽑아 결과를 넘겨준다. (그림 6)은 우리의 시스템을 실제 수행한 화면으로 우리의 시스템이 추구했던 목표에 맞게 상황에 따라 사용자 성향에 맞게 디바이스를 추천해줄 수 있었다.

(그림 6) 상황 기반 디바이스 추천 시스템 Result

5. 결론

본 논문에서는 유비쿼터스 환경에서 상황 기반의 디바이스 추천 시스템을 제안하였다. 우리의 시스템은 모바일환경에서 제한적인 자원 공유의 문제점을 해결할 수 있는 한 방법을 제시하였을 뿐만 아니라 상황에 따라 사용자 성향에 맞는 서비스를 해줄 수 있다는 점에서 의미가 있다고 할 수 있다. 향후에는 좀 더 사용자 성향에 가까운 서비스를 위한 지능화된 추론을 위한 연구를 할 것이며 USS 환경에 접했을 때, 모바일 디바이스가 스스로 요청할 수 있도록 확장에 대한 연구를 할 예정이다. 또한, 모바일 기기를 기반으로 하는 만큼 시스템의 경량화에 대한 연구를 할 것이다.

참고문헌

[1] M. Weiser, “Some computer science issues in ubiquitous computing,” Communications of the ACM, Vol. 36, Issue 7, pp.75-84, July 2003.
[2] 조위덕, “[유비쿼터스 컴퓨터 혁명] 유비쿼터스 컴퓨팅의 어제와 오늘,” 사이언스 타임즈.
[3] M. C. Mozer, “An intelligent environment must be adaptive,” IEEE Intelligent Systems and their Applications 14(2), pp. 11-13, May 1999.
[4] T. R. Halfhill, “Sun’s Jini: Science, Not Magic,” Micro processor report, pp.10-13, March 1999
[5] G. Bhatti, Z. Sahinoglu, K. A. Peker, J. Guo, and F. Matsubara, “A TV-Centric Home Network to Provide Unified Access to UPnP and PLC Domains,” Proceedings of the 2002 IEEE 4th International Workshop on Networked Applications, pp.234-242, Jan 2002.
[6] 박준희, 손영성 외, “홈 네트워크 미들웨어 및 표준화 동향,” ETRI, 전자통신동향분석 19권 5호, pp. 53-58, 2004. 10.
[7] JESS, the Rule Engine for the JavaTM Platform (http://herzberg.ca.sandia.gov/jess/)
[8] Maarten Menken. "Jess Tutorial", December 24, 2002
[9] SPARQL Query Language for RDF(http://www.w3.org/TR/rdf-sparql-query/)
[10] UPnP, http://www.upnp.org
[11] Jini, http://www.jini.org