논문 2009-2-18

유비쿼터스 컴퓨팅 환경에서 클러스터링 기반 효율적인 서비스 디스커버리 기법

Efficient Service Discovery Scheme based on Clustering for Ubiquitous Computing Environments

강은영*

Eun-Young Kang

요 약 유비쿼터스 컴퓨팅 환경에서는 네트워크상의 서비스를 자동으로 발견하고 서비스는 자신의 능력을 광고할 수 있는 서비스 디스커버리(service discovery)가 중요하다. 본 논문에서는 클러스터링 기반 서비스 디스커버리 기법과 P2P 캐쉥 기법을 혼합하여 효율적인 서비스 디스커버리 기법을 제안한다. 제안한 기법은 노드 ID를 기반으로 클러스터링 하고 이를 이용하여 서비스를 검색한다. 또한 서비스 검색을 빠르게 하기 위하여 P2P 캐쉬를 기반으로 이웃 노드의 정보를 사용하여 서비스 검색 성능을 향상시킨다. 제안된 기법은 노드의 부하를 가중 시키며 병목 현상을 일으키는 중앙 서버를 사용하지 않고 많은 쿼리를 생성하는 플러딩 방식을 사용하지 않는다. 시뮬레이션을 통하여 서비스 디스커버리를 이용하는 주고받는 메시지수를 줄이고 평균 탐색 거리를 줄임으로서 전체 네트워크 로드와 응답 시간이 성능 향상 면에서 우수함을 보인다.

Abstract In ubiquitous computing environments, service discovery to search for an available service is an important issue. In this paper, we propose an efficient service discovery scheme that is combined a node id-based clustering service discovery scheme and a P2P caching-based information spreading scheme. To search quickly a service, proposed scheme store key information in neighbor's local cache and search services using it's information. We do not use a central look up server and do not rely on flooding. Through simulation, we show that the proposed scheme improves the performance of response time and network load compared to other methods.

Key Words : Ubiquitous computing, Service discovery, Clustering

Ⅰ. 서 론

유비쿼터스 환경은 일반적인 클라이언트와 서버뿐만 아니라 셀폰, PDA, 노트북과 같이 움직일 수 있는 이동 컴퓨터들로 구성되며, 모바일 애드-혹 네트워크(Mobile Ad-Hoc Network: MANET)와 무선 LAN과 같은 무선 기반 네트워크의 결합으로 서로 연결되어 있다. 그와 같은 유비쿼터스 환경에서는 분산 시스템에 참여한 컴퓨팅 요소들은 시간에 따라 역동적으로 변한다. 사용자들은 자유롭게 네트워크에 참여하기도 하고 떠나가기도 하며 다른 사용자들과 계속적으로 위치가 변화하며, 디바이스들의 저장용량, 배터리 같은 자원들도 변화한다. 그러므로 유비쿼터스 컴퓨팅 환경에서는 네트워크상의 서비스를 자동으로 발견하고 서비스는 자신의 능력을 광고할 수 있는 서비스 디스커버리(service discovery)가 중요하다[1]. 아주 극심한 경우에, 유비쿼터스 환경은 모든 노드들이 모바일이고 그들의 위치가 역동적으로 변하는 모바일 애드-혹 네트워크를 포함한다[2].

*정회원, 인천대학교 컴퓨터공학과
접수일자 2009.3.23, 수정완료 2009.4.10

기존의 서비스 디스커버리 기법의 대표적인 프로토콜로는 SLP, UPnP, Jini 기법등이 있다. 그러나 Jini는 유선 환경에서 중앙 룩업 서버를 유지하는 중앙 디렉토리 기반 서비스 디스커버리 기법으로 중앙 서버에 부하가 집중되고 병목 현상과 같은 문제점이 있어 동적이고 자원이 부족한 모바일 애드혹 네트워크에는 적합하지 않다. UPnP, SLP2와 같은 프로토콜들은 802.11과 같은 무선 네트워크 기술을 도입하여 모바일 디바이스를 사용할 수 있도록 하였다. 그러나 아직은 모바일 환경보다는 유선 환경에 더 적합하도록 되어 있고 근본적인 문제 해결은 이루어지지 않았다. 또한 최근에 모바일 애드-혹에서 제한된 플러딩을 통한 서비스 검색을 위한 기법들이 제안되었으나 여전히 플러딩 기법을 사용하여 많은 쿼리 메시지를 생성하여 네트워크 비용이 크므로 자원이 부족하고 무선 환경인 모바일 애드-혹 네트워크에서는 효율적이지 않다.

본 논문에서는 모바일 애드-혹 네트워크의 제약사항들에도 불구하고 적절한 시간 내에 서비스를 검색, 사용할 수 있는 향상된 서비스 디스커버리 기법을 제안한다. 제안한 기법은 노드 ID를 기반으로 클러스터링하며, 이웃노드들의 정보를 캐시함으로써 많은 정보의 전달 효과와 이 정보를 이용함으로서 여행 단축 효과를 가진다. 제안한 기법은 노드수가 많은 네트워크에서도, 즉 확장성이 있고, 중앙 집중방식이 아닌 분산방식이며, 파일 탐색 시 주고받는 메시지의 수를 줄이고 평균탐색거리를 줄임으로서 전체 네트워크 통신비용과 응답 시간을 향상시킨다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 서비스 디스커버리을 위한 여러 가지 기법들을 살펴보고, 3장에서는 제안한 서비스 디스커버리에 대하여 설명하고, 4장에서는 제안한 방법의 성능평가에 대하여 설명하고, 5장에서는 결론을 맺는다.

Ⅱ. 서베스 디스커버리 기법

서비스 디스커버리 기법은 크게 두 가지로 구분된다. 첫째, 중앙 집중형 디렉터리 기반 서비스 탐색기법으로, 대표적인 프로토콜로는 Salutation[3], UDDI[4], Jini[5] 등이 있다. 이들은 중앙 룩업 서버를 유지하면서 이를 이용해 서비스 광고 및 서비스 탐색을 수행한다. 이러한 중앙 집중 방식은 중앙 룩업 서버 역할을 하는 특정 노드를 두어야 하므로 특정 기반 시설 없이 동작해야 하고 네트워크 특성이 동적인 모바일 애드-혹 네트워크에는 적합하지 않다. 또한 서버를 두더라도 서버에서 병목현상이 발생하기 때문에 노드들이 많은 대규모의 모바일 애드-혹 네트워크에는 적합하지 않다. 둘째, 분산 기법의 서비스 탐색으로, 대표적인 프로토콜로는 SLP[6], UPnP[7]등이 있으며, 이들은 멀티캐스트나 플러딩을 기반으로 하고 있다. 플러딩 방식은 중앙 룩업 서버를 사용하지 않는 분산 방식으로 동작하여 중앙 집중식 보다는 모바일 애드-혹 네트워크에 적당하다. 그러나 플러딩 기법은 많은 수의 쿼리 메시지를 생성하여 리소스가 제한적인 모바일 애드-혹 네트워크에서는 적당하지 않다.

모바일 애드-혹 네트워크에서 효율적인 서비스 탐색을 위하여 다양한 방법들이 제안되었다. 클러스터링은 네트워크에 서로 다른 관리 방식을 갖는 계층 구조를 두어 네트워크 유지와 파일 탐색에 필요한 오버헤드를 줄이는 방식이다. 클러스터링은 네트워크의 확장성을 보장하여 주기 때문에 오버해드에 대한 제약이 큰 모바일 애드혹 네트워크 환경에 적합하다. 최근 분산 기법중 클러스터링 기반으로 하는 기법들이 제안되고 있다. 대표적으로 확장성 있는 서비스 탐색 기법(SSD, Scalable Service Discovery)[8], 애드혹 망에서 효율적인 P2P 시스템[9], 모바일 애드-혹 네트워크에서 분산 해쉬 테이블 기반의 서비스 탐색 기법[10]등이 있다. 이 기법들은 분산된 클러스터링을 기반으로 디렉터리를 지원하는 기법이다. 분산된 서비스 디스커버리에서는 물리적인 계층과 분산된 클러스터 디렉터리를 지원하는 대표 노드들로 구성된 가상 계층의 2계층으로 나타내었다.

Ⅲ. 제안하는 서비스 디스커버리

3.1 협동적 정보 캐싱을 통한 이웃 노드 정보

서비스 제공자 노드는 자신이 제공하는 서비스에 대한 정보를 자신의 로컬 캐시에 저장한다. 또한 노드들이 인접해 있는 노드의 정보를 알 수 있다면 이러한 정보를 이용해 서비스 디스커버리의 효율성을 높일 수 있다. 본 논문에서는 이를 사용하기 위해 주기적으로 보내는 헬로우 메시지를 사용한다. 헬로우메시지는 무선 전송 범위 내에 있는 이웃들에게 주기적으로 한-홉 브로드캐스트

한다. 각 노드의 이웃노드 서비스 정보를 알기 위하여 헬로우 메시지에 노드의 ID, 서비스 정보를 포함하여 전송한다. 그러므로 따로 이웃노드의 정보를 얻기 위한 메시지를 사용하지 않는다. 이웃노드로부터 받은 헬로우 메시지를 기반으로 이웃노드의 존재와 그의 정보를 인접노드리스트에 저장한다. 이웃노드로부터 일정기간 헬로우 메시지를 받지 못하면 이웃노드가 이동하여 더 이상 이웃하지 않음을 의미하므로 인접노드리스트에서 그 노드는 제외하면 된다. 이렇게 저장된 정보들은 다른 노드들이 서비스 정보를 필요로 할 때 유용하게 사용될 수 있다. 왜냐하면 서비스를 필요로 하는 노드가 서비스를 제공하는 서비스 제공자나 대표 노드에게 까지 도달하지 않고도 이 캐시된 정보를 이용하여 응답하게 함으로써, 노드가 여행하는 경로의 수가 줄어들어 여행 단축 효과가 있어 전체 네트워크 로드를 줄일 수 있다.

3.2 시스템 개요

본 논문에서 제안하는 노드 ID 기반 클러스터링 기법은 몇 개의 클러스터들로 구성될 수 있다. 각 클러스터는 3가지 종류로 구성되는데 각각 클러스터 헤더, 게이트웨이 노드, 일반 노드로 들 수 있다.

클러스터에서 각 노드는 3가지 중 하나의 역할을 담당한다. 메시지를 전달하려고 시도하는 노드는 메시지를 전달할 노드로 주변 노드 중 노드 ID가 자신의 노드 ID보다 작은 노드를 선택하고 메시지를 전달한다. 이러한 메시지 전달 정책으로 메시지는 노드 ID가 가장 낮은 노드들로 전달되며, 가장 낮은 노드 ID를 가진 노드에게 전달되면 더 이상의 메시지 전달은 발생하지 않는다.

메시지가 마지막으로 전달된 노드를 '클러스터 대표' 노드라 한다. 클러스터 대표 노드는 네트워크에 여러 개가 존재할 수 있다. 어떤 노드가 자신의 노드 ID 보다 더 낮은 노드 ID를 가지는 이웃 노드가 복수개가 존재하는 경우 클러스터 헤더는 중첩되고 이 노드는 두 클러스터에 속하게 된다. 이 노드를 '게이트웨이 노드' 라 한다. 클러스터의 생성은 메시지 전달에 따라 자동으로 구성되며, 각 이동 노드는 자신의 노드 ID와 이웃노드관리테이블에 있는 인접 노드의 노드 ID를 비교함으로서 자신이 클러스터 대표 노드인지 게이트웨이 노드인지를 판단한다. 또한 노드의 이동에 따라 클러스터 대표 노드와 게이트웨이 노드는 동적으로 변경될 수 있으며, 클러스터 역시 동적으로 변하게 된다. <그림 1>은 애드-혹 네트워크에서 클러스터링 과정의 예를 보여준다. <그림 1>에서 (a)는 클러스터링 생성 전의 애드-혹 네트워크를 나타내고, (b)는 메시지 전송 후 클러스터링이 생성된 모습을 나타낸다.

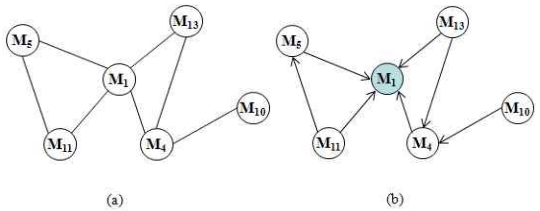

그림 1. 클러스터 과정
Fig. 1. Cluster Formation Processing

노드 M_5는 자신과 이웃하고 있는 노드 M_1, M_{11}모두에게 메시지를 보내지 않고 자신의 노드 ID보다 작은 노드 ID를 가진 노드 M_1에게만 메시지를 보낸다. 노드 M_{11}은 자신과 이웃하고 있는 노드 M_5, M_1이 모두 자신보다 낮은 노드 ID를 가지고 있으므로 메시지를 보낸다. 노드 M_4는 이웃 노드 M_1, M_{10} 노드 중 자신보다 낮은 노드 ID를 가지고 있는 M_1에게만 메시지를 보낸다. 노드 M_1은 자신보다 낮은 노드 ID를 가진 이웃 노드가 없으므로 메시지 전달은 더 이상 발생되지 않는다. 그러므로 M_1은 클러스터 대표 노드가 된다. 이런 과정을 되풀이 하여 클러스터링이 생성된다.

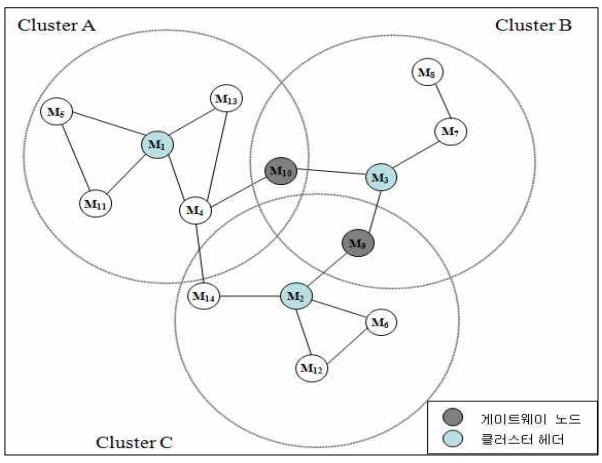

그림 2. 모바일 애드-혹 네트워크에서의 클러스터링 예
Fig. 2. Example of Clustering in a Mobile Ad hoc Network

그림 2에서 이와 같은 과정을 반복하면 노드 M_3, M_7, M_8, M_9, M_{10}은 클러스터B의 형태로, 노드 M_2, M_6, M_9,

M_{12}, M_{14}는 클러스터 C의 형태를 구성한다. 이때, 양쪽 클러스터에 속한 노드 M_9과 M_{10}는 게이트웨이 노드로서, 그들은 클러스터들 간의 메시지 전달 역할을 한다.

3.3 서비스 디스커버리

서비스를 탐색하고 싶을 때 먼저 노드는 자신이 속한 클러스터에 있는 클러스터 대표 노드에게 서비스 요청 메시지를 보낸다.

클러스터 대표 노드는 클러스터 내에 존재하는 멤버 노드들에 대한 파일 정보와 재탐색을 위한 캐시를 가지고 있으므로 탐색 요청 정보에 해당하는 파일 정보가 있으면 이에 대해 응답 메시지를 보낸다. 만약, 파일 요청 메시지가 클러스터 대표 노드에게까지 전달되고도 해당 서비스 정보를 발견하지 못했을 경우, 클러스터 대표 노드는 다른 클러스터 대표 노드에게 메시지를 보낸다. 클러스터간의 메시지 전송을 통하여 요청 파일을 검색하는 것이다. 클러스터간의 메시지 전송은 게이트웨이 노드가 담당하게 된다. 결국 자신이 속한 클러스터 대표 노드나 다른 클러스터에서 해당 서비스 요청 과정을 반복하게 되고 요청한 서비스를 발견하게 된다. 다음은 서비스 탐색 과정에 대한 알고리즘을 <알고리즘 1>에서 보인다.

```
//서비스 요청 메시지가 도착 했을 때
Request_Search(Mi)
{
  For every data entry list in local cache
  {
        if  requested data exists in local cache
            send response message to data
            requester
 }
/* 노드 ID를 기준으로 이웃 노드중 전송할
 이웃노드를 선택, 이웃노드관리테이블
  (NNMT) */
  while (k < size of NNMT) {
       For each neighbor n in NNMT {
           if  requested data exists
                 in NNMT[k]
               send response message
                 to data requester
           else
               forward Request_Search(Mi)
                  to 클러스터 대표 노드
         }
       k = k + 1;
  }
}
```

알고리즘 1. 서비스 탐색 프로토콜
Algorithm. 1. Service Discovery Protocol

3.4 캐시 갱신 기법

모바일 애드-혹 네트워크에서 대역폭과 전력 제한 때문에 강한 캐시 일관성을 유지하는 것은 너무 많은 오버헤드가 있다[11]. 그러므로 캐시 일관성을 유지하기 위하여 본 논문에서는 TTL 메커니즘을 기반으로 한다. 캐시된 데이터 엔트리의 서비스 가능 시간이 초과되지 않았다면 적정한 값으로 간주한다.

Ⅳ. 실험 및 결과

이번 장에서 본 논문에서 제안하는 서비스 프로토콜의 성능을 평가한다. 본 논문에서 제안하는 프로토콜(Lower-CL)을 기존의 플러딩 방식(Flood-Based)과 비교한다.

4.1 성능 평가 환경

이동 애드-혹 네트워크에서 서비스 디스커버리의 성능평가를 위해 대표적인 네트워크 시뮬레이션 도구인 NS2 v2.32을 사용하였다. 표 1은 알고리즘을 위한 시뮬레이션에 사용된 시스템 환경 및 인자의 값들을 보인다.

표 1. 시스템 환경
Table 1. System Environment

환경변수	값
노드 수	80 개
영역 크기	1000 X 1000 m
평균이동속도	3 ~ 20 m/s
정지시간	2 s
이동성 모델	Random Way Point
전송거리	250 m
MAC 프로토콜	802_11
최대이동속도	5 ~ 30 m/s
시뮬레이션 시간	250 s
평균 쿼리 요청 시간	5 s

4.2 성능 평가 결과

본 논문에서 성능 평가 기준으로 다음과 같은 3가지가 사용된다. 첫째, 평균 검색 거리는 검색에 소요된 평균 홉 수로 구한다. 둘째, 네트워크 로드 서비스 광고 메시지 수와 서비스 검색 메시지 수를 합하여 구한다. 셋째, 검색

성공률은 생성된 전체 질의 중에서 서비스 검색을 성공한 질의의 비율이다.

<그림 3>은 최대 이동 속도에 따른 평균 탐색 거리를 모의 실험한 결과이다. 대부분의 경우 이웃한 노드의 로컬 캐시에 있는 정보를 이용하여 서비스를 찾게 되므로 0.5 ~ 2홉 사이에서 원하는 서비스를 찾는다. 0홉에서 찾는다는 것은 본인이 가지고 있는 로컬에서 원하는 정보를 찾는다는 것을 의미한다. 그림에서 나타내듯이 제안하는 기법들은 대부분 2홉 이내에 있는 클러스터 대표 노드나 이웃 노드에서 원하는 서비스를 찾고 플러딩 기법에서는 10홉이 넘은 거리에서 원하는 서비스를 찾는다. 본 논문에서 제안한 방식이 플러딩 기법보다 평균 탐색 거리가 가까워 빠른 검색 결과를 보여 좋은 성능을 보인다. 이는 서비스 요청자가 서비스 요청 메시지를 보낸 후 서비스 응답 메시지를 받을 때까지의 거리로 거리가 짧을수록 검색에 빨리 성공했으며, 주고받는 메시지 수도 적어 진다.

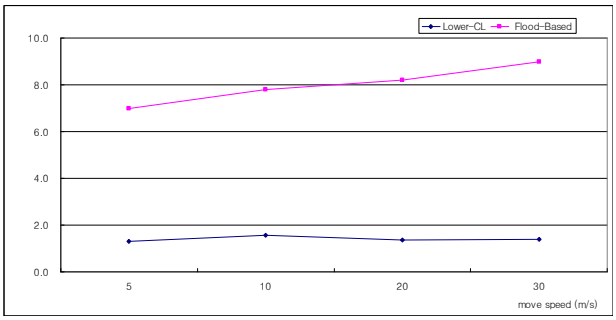

그림 3. 평균 탐색 거리
Fig. 3. Average Search Distance

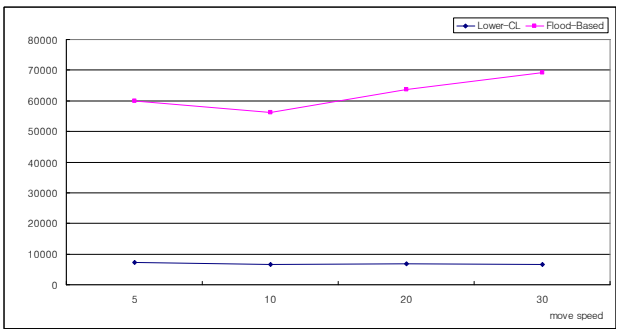

그림 4. 네트워크 로드
Fig. 4. Network Loads

<그림 4>는 노드 이동 속도에 따른 네트워크 로드를 보인다. 플러딩 기반 서비스 디스커버리 기법은 서비스 광고 메시지를 보내지 않는다. 플러딩 기법은 네트워크에 있는 모든 도달 가능한 노드에게 탐색 요청 메시지를 보낸다. 플러딩 기법에서 중복 되게 받은 메시지는 다시 보내지 않고 폐기 하며, 새로 받은 메시지만 탐색 요청을 위해 전달한다. <그림 4>에서 나타내듯이 노드의 이동 속도가 증가 할수록 플러딩 기법에서는 네트워크 로드가 늘어나나, 제안하는 기법은 노드의 이동 속도가 늘어나더라도 네트워크 로드가 증가하는 율은 적어 좋은 성능을 나타냈다. 이는 서비스 제공자가 서비스 광고 메시지를 보낸 후, 이를 받은 d-hop 이웃 노드들은 이를 캐시하여 서비스 검색 요청을 받았을 때, 유용하게 사용될 수 있어 네트워크 로드를 줄일 수 있다.

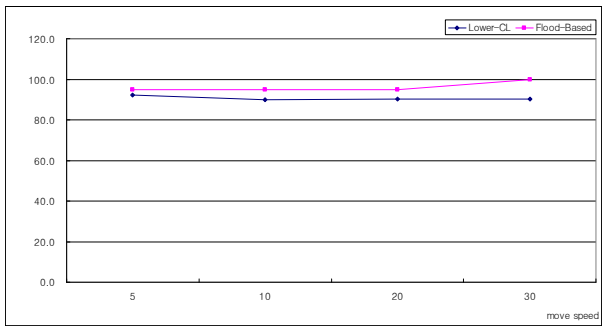

그림 5. 서비스 검색 성공률
Fig. 5. Service Search Success Rate

<그림 5>는 각 프로토콜별로 노드 이동 속도에 변화를 주어가며 검색 성공률을 모의 실험한 결과이다. 본 논문에서 제안한 방식이 서비스 검색 성공률이 높고, 플러딩 기법보다는 성공률이 낮음을 보인다. 그러나 대부분의 노드 수에서 성공률은 95%이상을 보인다. 만일 서비스 광고 시 서비스에 대한 정보가 네트워크상에 효율적으로 캐시 된다면, 검색 성공률이 높아질 것이다.

V. 결 론

본 논문은 모바일 애드-혹 네트워크에서 노드ID기반 클러스터링을 통한 효율적인 파일 탐색 기법을 제안한다. 기존의 서비스 탐색 기법 중 중앙 집중식 탐색방식과 플러딩 방식은 여러 가지 자원이 부족한 모바일 애드-혹 네트워크에 적합하지 않다. 중앙 집중 방식은 서버의 집중 문제와 플러딩 방식의 서비스 탐색은 많은 쿼리를 발생한다. 본 논문에서는 노드 ID기반의 클러스터링을 통하여 파일 정보를 요청 시 여행해야할 경로를 줄여 평균

탐색 거리와 전체 네트워클 로드를 줄인다. 기존의 플러딩 방식 서비스 탐색기법보다 성능이 우수함을 모의실험을 통해 보였다.

참 고 문 헌

[1] D. Chkraborty, A. Joshi, Y. Yesha, “Toward Distributed Service Discovery in Pervasive Computing Environments”, IEEE Trans. on Mobile Computing, Vol. 5, No. 2, pp. 97-112, Feb 2006.

[2] C.K. Toh, “*Ad Hoc Mobile Wireless Networks: Protocols and Systems*”, Prentice Hall, 2002

[3] The Salutation Consortium Inc., "Salutation Architecture Specification", *http://www.salutation.org*

[4] “Universal Description Discovery and Integration Platform", http://www.uddi.org

[5] K. Arnold, B. Osullivan, R.W. Scheifler, J. Waldo, and A. Wollrath, "*The jini Specification (The Jini Technology)*", Reading, Mass, Addison-Wesley, June 1999

[6] E. Guttman, C. Perkins, and J. Veizades, RFC 2165, "Service Location Protocol", June 1997

[7] R. John, "UPnP, Jini and Salutation - A Look at Some Popular Coordination Frameworks for Future Network Devices", technical report, California Software Labs, 1999

[8] F. Sailhan and V. Issarny, "Scalable Service Discovery for MANET", IEEE PERCOM, pp.235-244, 2005

[9] 최현덕, 박호현, 우미애, "애드혹 망에서 효율적인 P2P 시스템", 한국통신학회, 제32권, 제4호, 200-207쪽, 2007년 4월

[10] 정재훈, 이승학, 김남기, 윤현수, "모바일 애드-혹 네트워크에서 분산 해쉬 테이블 기반의 서비스 탐색 기법", 한국정보과학회, 제35권, 제1호, 91-97쪽, 2008년 2월

[11] G. Cao, L.Yin and C. Das, “A Cooperative Cache Based Data Access Framework for Ad Hoc Networks”, IEEE Computer, Feb. 2004

저자 소개

강 은 영(정회원)

- 제8권 제6호 참조
- 2009년 2월 : 성균관대학교 컴퓨터공학부 공학박사
- 2009년 2월 ~ 현재 : 인천대학교 박사후연구원

<주관심분야> Ubiquitous computing, Service discovery, Clustering