논문 13-38C-12-06

한국통신학회논문지(J-KICS) '13-12 Vol.38C No.12
http://dx.doi.org/10.7840/kics.2013.38C.12.1134

참가자가 밀집된 환경에서의 게재/구독을 위한 분산 해쉬 기반의 고속 서비스 탐색 기법

안 시 내◆, 강 경 란°, 조 영 종*, 김 노 원*

Distributed Hashing-based Fast Discovery Scheme for a Publish/Subscribe System with Densely Distributed Participants

Si-Nae Ahn◆, Kyungran Kang°, Young-Jong Cho*, Nowon Kim*

요 약

Pub/sub 기술은 메시지 기반으로 데이터 생성자의 위치, 시간, 동기화 등에 대한 데이터에 제한 없이 접근할 수 있게 하는 특징을 가지므로, 데이터 중심 서비스를 위한 미들웨어 구축 기술로 널리 활용되고 있다. 국제 표준화 기구인 OMG (Object Management Group)에서 정의한 DDS (Data Distribution Service)는 pub/sub 기반의 미들웨어 기술로서, 미국 군용 장비의 표준 미들웨어로 채택되는 등 그 유용성이 높이 인정받고 있다. 그러나 publisher와 subscriber가 밀집된 환경에서는, 시스템 초기 부팅 시에 시스템 내 data 생산과 소비 주체가 되는 Participant와 Endpoint들을 탐색하는 과정에서의 지연 시간이 길다는 문제점을 갖고 있다. 본 논문에서는 지역적으로는 넓지 않지만 시스템 내의 Participant와 Endpoint의 수가 밀집된 환경에서의 탐색 시간을 줄일 수 있는 방안을 제시한다. 기존의 DDS 표준에서 정의하고 있는 표준 탐색 단계인 Participant 탐색 단계와 Endpoint 탐색 단계를 통합하고 분산 해쉬 기법의 Successor 개념을 도입하여 각 Participant마다 메시지를 전달해야 하는 대상의 수를 줄였다. 메시지 전달 대상의 수를 줄임으로써 전송 프로토콜로 TCP를 적용하는 것이 가능해져, 메시지 전달의 신뢰성을 높일 수 있었다. 네트워크 시뮬레이터를 통한 성능 평가에서 본 연구에서 제안한 기법이 기존 기법에 비해 10%의 탐색 시간으로 시스템 내 Participant와 Endpoint를 발견할 수 있었다.

Key words : publish/subscribe system, middleware, distributed hash, DDS, peer discovery

ABSTRACT

Pub/sub system enables data users to access any necessary data without knowledge of the data producer and synchronization with the data producer. It is widely used as the middleware technology for the data-centric services. DDS (Data Distribution Service) is a standard middleware supported by the OMG (Object Management Group), one of global standardization organizations. It is considered quite useful as a standard middleware for US military services. However, it is well-known that it takes considerably long time in searching the Participants and Endpoints in the system, especially when the system is booting up. In this paper, we propose a discovery scheme to reduce the latency when the participants and Endpoints are densely distributed in a small area. We propose to modify the standard DDS discovery process in three folds. First, we integrate the Endpoint discovery process with

※ 본 연구는 지식경제부 및 한국산업기술평가관리원의 산업원천기술개발사업의 일환으로 수행하였음. [10035708, 고신뢰 자율제어 SW를 위한 CPS(Cyber-Physical Systems) 핵심 기술 개발]
◆ First Author : 아주대학교 컴퓨터공학과 컴퓨터통신연구실, kacey87@naver.com, 학생회원
° Corresponding Author : 아주대학교 컴퓨터공학과 컴퓨터통신연구실, korykang@ajou.ac.kr, 정회원
* 아주대학교 컴퓨터공학과 컴퓨터통신연구실, yjcho@ajou.ac.kr, 종신회원, rowony@naver.com
논문번호 : KICS2013-04-170, 접수일자 : 2013년 4월 10일, 심사일자 : 2013년 6월 5일, 최종논문접수일자 : 2013년 12월 6일

the Participant discovery process. Second, we reduce the number of connections per participant during the discovery process by adopting the concept of successors in Distributed Hashing scheme. Third, instead of UDP, the participants are connected through TCP to exploit the reliable delivery feature of TCP. We evaluated the performance of our scheme by comparing with the standard DDS discovery process. The evaluation results show that our scheme achieves quite lower discovery latency in case that the Participants and the Endpoints are densely distributed in a local network.

Ⅰ. 서 론

Publish/subscribe (Pub/sub) 기술은 데이터 교환에 참여하는 사용자들이 데이터 생산자 (Publisher)나 데이터 소비자 (Subscriber)가 되어 데이터를 생성하여 배포하고 원하는 데이터를 소비하는 데이터 배포 기술이다. Pub/sub 기술은 메시지 기반으로 데이터 생성자의 위치, 시간, 동기화 등에 대한 제한 없이 데이터에 접근할 수 있게 하는 특징을 갖는다. Pub/sub 기술을 데이터 전달 기법으로 적용하고 있는 미들웨어들이 다수 상용화 되어 있으나, 그 중 DDS (Data Distribution Service) [1]는 국제적인 표준화기구인 OMG (Object Management Group)를 중심으로 표준화된 기술로서 미국 등에서 군용 장비의 표준 미들웨어로 장착되고 있다. 국내에서도 대규모 함정 시스템 등을 개발하는 미들웨어로 사용되고 있으며, 사이버 물리 시스템 (Cyber Physical System, CPS)를 구현하는 기술의 일환으로 주목받고 있다[2].

DDS 표준에서 Publisher와 Subscriber 사이의 신뢰성 있는 데이터 송수신과 다양한 품질의 송수신 서비스를 제공하기 위한 목적으로 시스템 내 Participant와 Participant 내 Endpoint, 즉 Data Reader와 Data Writer에 대한 전체 정보를 모든 Participant가 갖고 있어야 한다고 요구하고 있다[3]. 즉 DDS가 장착된 시스템이 구동하기 시작하는 시점부터 시스템 내의 전체 Participant와 Endpoint들을 탐색하는 절차를 마친 후에야 실제 데이터 송수신이 가능해진다. 시스템 부팅 단계에서 Endpoint가 생성되는데 걸리는 지연 시간을 고려하여 Participant를 탐색 절차와 Endpoint를 탐색 절차가 분리되어 있다. 추후에 시스템 내에서 탈퇴하거나 시스템에 새로 가입하는 Participant나 Endpoint들에 대해서 타 Participant들과 Endpoint들에게 자신의 참여 상태 변화를 등록해야 한다. DDS가 추구하는 신뢰성 있는 데이터 전달과 데이터 전달의 품질 보증을 위해 이러한 서비스 개시 전 탐색 절차가 필수적이지만, Participant의 수가 늘어나고 Endpoint의 수가 늘어나게 되면 탐색 과정에서 발생하는 메시지의 수와 탐색을 마치기까지 소요되는 시간이 시스템의 성능을 저하시킬 정도로 커지게 된다. 그리고, Participant 탐색 절차와 Endpoint 탐색 절차가 분리되어 있어 각 절차의 신뢰성 보장을 위한 과정에서 발생하는 메시지 부담이 크다.

본 논문에서는 시스템 부팅 단계에서 기존의 DDS의 탐색 과정에서 발생하는 메시지의 양과 탐색 완료에 걸리는 시간을 획기적으로 줄이기 위한 기법을 제시한다. 본 논문에서 제안하는 기법은 Participant들 간의 오버레이 네트워크를 구축하는 기법[4]과 DHTbroadcast를 적용하는 기법[5]으로 구성되어 있다. 본 논문에서 제시하는 기법의 구체적인 단계는 DDS의 표준 명세에 밀접하게 관련이 있으

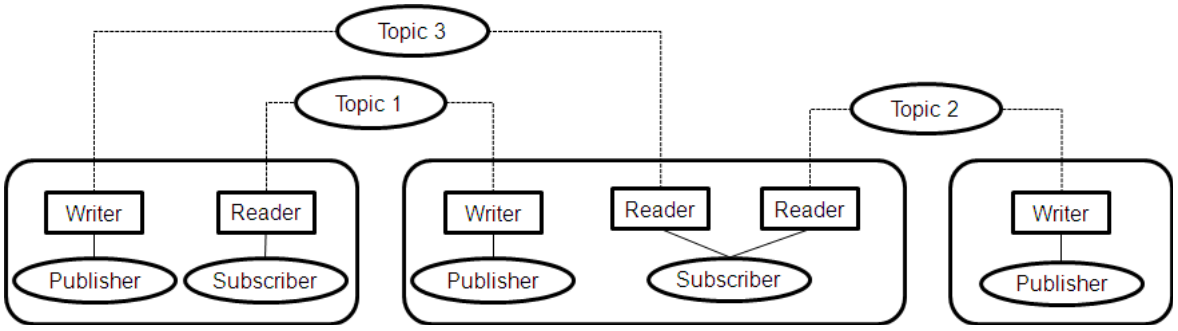


그림 1. DDS 내 개체들의 관계
Fig. 1. The relationship among the DDS entities

나, 기본 알고리즘은 참여 노드에 대한 완벽한 정보를 획득해야 하는 것을 요구하는 Pub/sub 시스템이라면 어떠한 것에도 적용 가능하다.

본 논문에서 제시하는 기법이 갖는 기술적인 특징은 크게 세 가지로 정리할 수 있다 첫째, 기존의 DDS 표준에서는 Participant에 대한 정보와 각 Participant 내 Endpoint 정보가 다른 절차를 통하여 전달되지만, 본 논문에서는 Participant가 자신이 보유한 Endpoint들의 정보를 하나의 메시지에 포함시켜서 전송한다. 둘째, 기존 연구[6]에서 제시하는 DHT 기반 브로드캐스트 기법을 사용하여 메시지를 전체 시스템 내의 Participant들에게 확산함으로써, 각 참여 Participant가 메시지를 전달하는 대상의 수를 감소시키고 네트워크 내에서 전송되는 메시지의 수를 감소시킨다. 셋째, DHT 기반 브로드캐스트 기법에서, 메시지의 중간 전달 단계에서의 Participant들 간에 TCP를 사용하여 메시지를 전송함으로써 시스템 내의 모든 Participant로의 신뢰성 있는 메시지 전달을 보장한다.

본 논문에서 제안하는 기법과 기존의 DDS 표준에 명시된 탐색 기법을 '탐색 완료 시간'과 '메시지 전달율'이라는 관점에서 시뮬레이션을 통해 성능을 비교하였다. '탐색 완료 시간'은 각 노드가 전체 Participant와 Endpoint의 정보를 확보하는데 소요되는 시간을 의미하고, '메시지 전달율'은 탐색 프로토콜을 위해 발송된 제어 메시지의 양과 실제 수신된 메시지의 양을 비율을 의미한다. 예상한 바와 같이, 본 논문에서 제안하는 기법이 높은 메시지 전달율을 보였으며, 기존 기법에 비해 탐색 시간을 현격하게 감소시키는 것을 관찰할 수 있었다.

본 논문의 구성은 다음과 같다. 제 2장에서는 본 논문의 배경인 DDS 표준에 대한 기술한다. 제 3장에서는 본 논문에서 제안하는 기법을 구체적으로 기술하고, 제 4장에서는 제안한 프로토콜을 시뮬레이터로 구현하여 성능 분석한 결과를 기술한다. 제 5장에서 결론과 향후 연구 계획을 제시하며 논문을 마친다.

Ⅱ. 관련 연구

DDS는 Topic 기반의 Pub/sub 통신 모델을 기초로 하는 대표적인 미들웨어이다. DDS 미들웨어를 사용하는 Application은 Domain Participant를 통해 통신이 이루어진다. Domain Participant에는 데이터 송신을 위한 Publisher와 수신을 위한 Subscriber가 있다. 데이터의 종류를 구분하기 위한 Topic이 있어 같은 Topic을 가지는 Publisher와 Subscriber사이에서 통신이 이루어진다. 즉 특정 Topic에 대하여 생성된 데이터는 Publisher를 통해서 송신되고 해당 Topic을 수신하고자 하는 Subscriber에 전달된다. DDS가 정상적으로 동작하기 위해 이러한 요소들이 우선적으로 탐색이 되어야 한다. 즉 타 Participant 정보를 우선 탐색하고, Data Reader와 Data Writer들의 정보를 서로 탐색하는 Discovery protocol이 수행되어야 한다.

서비스 탐색 프로토콜은 다음의 두 가지 요소에 대한 탐색을 하는 것이 목적이다. 첫째, 각 Domain Participant는 네트워크에 존재하는 모든 타 Participant를 탐색해야 한다. Participant를 탐색하는 과정은 네트워크에 존재하는 Participant를 탐색하는 과정이다. 둘째, 탐색한 모든 타 Participant가 가지는 Data Reader와 Data Writer에 대한 정보를 탐색해야 한다. 이는 타 Participant가 Pub/sub하는 Topic 정보를 교환하는 과정이다. 데이터 통신을 수행하기 위한 사전 작업으로 Topic을 가진 Publisher와 Subscriber에 대한 탐색 과정이다. DDS가 서비스 탐색 프로세스에서 사용하는 기본 서비스 탐색 기법을 SPDP(Simple Participant Discovery Protocol)과 SEDP(Simple Endpoint Discovery Protocol)라고 한다. 모든 Participant가 네트워크의 모든 서비스 탐색을 완료해야 데이터 교환이 이루어지게 된다.

그림 2는 DDS 표준에서 SPDP와 SEDP를 통한 서비스 탐색 프로세스를 보여주고 있다. SPDP과정에는 Participants Discovery, Handshaking, SEDP에는 Endpoint Discovery, Liveness의 단계가 있다. Participants Discovery과정에서 Participant data는 Multicast로 전송된다. 처음 참여 시에 6회 전송 후, 주기적으로 Participant data를 전송하여 나중에 참여하는 Participant도 자신의 Participant data를 수신할 수 있도록 한다. 처음 참여 시에 다수개의 Participant data를 전송하는 것은 Participant data에 대한 Reliability를 올리기 위해서이다.

Participant가 타 Participant에게 발견했음을 알리기 위한 Handshaking과정이 있다. Participant data를 송신한 Participant는 타 Participant로부터 HeartBeat 메시지를 받음으로서 타 Participant가 자신을 탐색했음을 알 수 있다. 이후 해당 타 Participant에게 ACK를 보내 Handshaking 과정을 마친다.

PDP 과정이 끝나면 EDP 과정이 시작된다. 서로 탐색이 완료된 두 Participant들은 자신들이 보유한

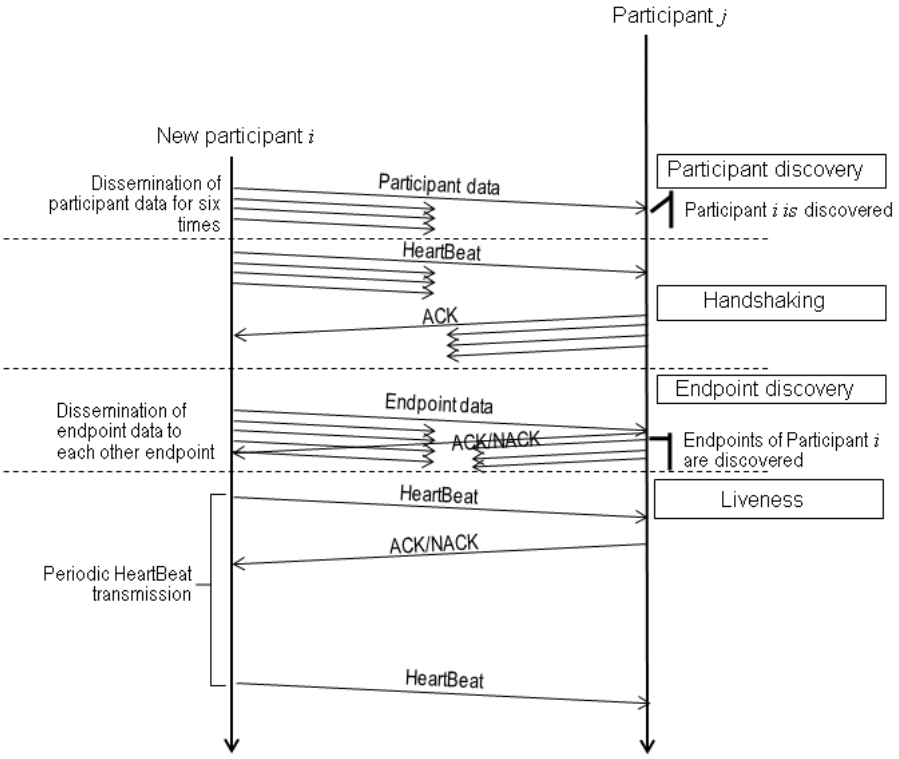


그림 2. DDS 표준에서의 탐색 절차
Fig. 2. Discovery process defined in the DDS Standards

각 Endpoint, 즉 Data Reader와 Data Writer들에 대한 정보를 담은 Endpoint data 메시지를 상대방에게 전송한다. Endpoint data 메시지를 수신하면 이에 대해 ACK 메시지가 상대방에게 전송해야 한다.

*e*개의 Endpoint를 보유한 Participant P는 Participant Q에게 *e* 개의 Endpoint_data 메시지를 발송하고 Q로부터 *e* 개의 ACK을 수신해야 Q가 P의 Endpoint 발견을 완료한 것이 된다. 만일 Participant의 수가 *n*이고 각 Participant가 총 *e* 개의 Endpoint, 즉 총 *e* 개의 Data Reader와 Data Writer를 갖고 있다면, 이 시스템에서의 EDP 과정을 모두 마치려면 각 Participant가 $e\times(n-1)$개의 Endpoint_data 메시지를 전송하고 $e\times(n-1)$ 개의 ACK을 수신해야 하며 $e\times(n-1)$ 개의 Endpoint_data 메시지를 수신하고 $e\times(n-1)$ 개의 ACK을 전송해야 한다. 즉 *n* 개의 Participant가 존재하는 DDS 시스템에서 Participant가 보유한 최대 Endpoint의 수가 e라고 할 때, 메시지 교환 횟수라는 관점에서 $O(k\times e\times n)$의 복잡도를 갖는다고 할 수 있다.

기존의 SDPBloom[7]은 각 Participant가 교환해야 하는 탐색 과정에서 교환해야 하는 메시지의 크기를 줄이고, 모든 Endpoint 들의 정보를 관리하기 위해 필요한 메모리의 양을 절약하기 위한 방안을 제시한다. 기존의 SDP가 타 Participant들이 가진 Endpoint 정보를 모두 요청했다면, SDPBloom은 블룸 필터를 이용하여 타 Participant가 이미 확보한 Endpoint 정보들이 무엇이 있는지 알아낸 후에 필요한 정보만을 추가적으로 요청한다. 각 Participant가 자신이 확보한 Endpoint 정보를 표현할 때, 필요한 Endpoint 정보를 표현할 때 블룸 필터를 사용한다. 전체적으로 교환해야 하는 메시지의 크기를 크게 감소시킴으로써 탐색 완료 시간 역시 기존 기법에 비해 감소시킬 수 있다. 그러나 여전히 UDP를 사용하여 각 Participant가 모든 나머지 Participant와 메시지를 교환해야 하는 부담은 존재한다.

Ⅲ. 제안하는 기법

3.1. 메시지 확산 기법

본 논문에서 고려하는 DDS 적용 환경은, 좁은 지역에 밀집된 단말들이 다양한 종류의 정보를 생성하고 소비하는 상황이다. 특히, 함정과 같이 제한된 공간 내에 다수의 장비들이 설치되어 있고, 작전 수행을 위해 다수의 장비들이 다수의 나머지 장비들로부터 데이터를 수집해야 하는 상황을 가정하고, DDS의 초기 탐색 시간을 줄이기 위한 방안을 제시

한다.

좁은 지역, 즉 하나의 스위치 혹은 하나의 라우터 하에 다수의 단말들이 네트워크를 구성하는 환경을 가정하므로, 탐색 시간을 증가시키는 요소로 전달 지연 시간 (Propagation delay)를 고려하지 않는다. 오히려, 동시에 다수의 메시지 전송이 시도되어 매체 접속 시의 충돌이 발생할 수 있다. 특히, Participant 탐색 과정에서는 모든 단말이 동일한 멀티캐스트 그룹에 가입하므로 IGMP snooping[8]이 아무런 효과를 발휘하지 못하고 하나의 단말이 보낸 메시지가 네트워크 전체로 확산되게 되므로 다수의 단말이 동시에 메시지를 발송하게 되면 멀티캐스트 스톰이 발생하여 메시지가 손실되고 이에 따라 탐색 과정이 지연될 수 있다. 본 논문에서는 이러한 상황을 특히 중요하게 고려하여 동시에 메시지를 전송하는 단말의 수를 줄이면서도 메시지 전달 효과를 높이는 방안을 제시하고자 한다.

기존 DDS의 탐색 기법에서는 Participant를 탐색하는 과정과 Endpoint를 탐색하는 과정이 분리되어 있다. 또한, Endpoint 탐색 시에는 각 Participant 내의 탐색 Endpoint가 나머지 Participant들의 탐색 Endpoint들과 일대일로 메시지를 송수신하여 탐색 과정이 진행된다. 이 두 가지 이유로 인해 기존 DDS 표준에서의 기법은 탐색 과정에서 발생하는 메시지의 수가 많아지고 지연 시간이 길어진다. 본 논문에서 제안하는 기법은 탐색 과정에서 Participant와 Endpoint를 탐색하는 과정을 통합하고, 일대일로 메시지를 송수신하는 대상의 수를 줄이는 방안을 제시한다.

Participant와 Endpoint들을 탐색하는 과정을 통합하기 위해, 기존 DDS 명세에서 정의한 Participant 정보와 Endpoint 정보 외에 자신이 발견한 Participant와 Endpoint들을 표시하는 탐색 비트맵을 추가로 정의한다. 시스템에 가입하는 Participant는 Participant data, Endpoint data와 탐색 비트맵을 포함한 메시지를 시스템 내 타 Participant들에게 전송한다. 앞으로는, 개별 Participant의 Participant data와 Participant 내의 Endpoint들에 대한 Endpoint data를 통합하여 MAP이라 칭한다. 실제 데이터 메시지를 송수신하는 것은 Endpoint들이므로 기존의 DDS 탐색 기법에서 Participant 탐색 과정을 변형하여 Participant 정보와 Endpoint 정보까지 함께 수집할 수 있도록 한다.

부팅 단계에서 Endpoint들의 생성에 지연 시간이 발생하여 제안하는 기법의 탐색 과정에서 누락되는 경우가 발생할 수 있다. 그러나, DDS에서의 탐색 절차는 일시에 끝나는 것이 아니고 주기적으로 그리고 필요에 따라 재개되는 절차이므로 누락된 Endpoint들의 정보를 따로 관리하고 있다가 다음 주기에 타 Participant들에게 통지하여 발견되도록 한다.

본 논문에서는 일대일로 메시지를 송수신하는 대상의 수를 줄이기 위해, 기존에 구조화된 p2p 네트워크에서의 메시지 확산을 위한 연구[6]에 제시된 기법을 본 연구의 환경에 맞게 변형한 기법[5]을 활용한다. Chord[9]의 Finger Table과 유사한 Successor List를 활용한다. Successor는 각 Participant의 id에 비해 2^0, 2^1, .., 2^j로 증가하는 id를 갖는 Participant를 의미한다. 해당 Participant의 id가 i일 때, j번째 Successor는 $i+2^j$의 id를 가질 것이며, $(i+2^j,\ i+2^{j+1}-1)$ 범위에 해당하는 Participant들에게 메시지를 전달하는 것을 담당한다고 간주한다. 시스템 내의 Participant가 가질 수 있는 최대 id가 maxID라고 할 때, Successor List의 행의 갯수는 최대 $\log_2$maxID이 된다.

각 Participant는 순차적으로 시스템에 가입하는 것이 아니고, 임의의 순서로 가입을 하거나 어떤 외부의 다른 이유로 탈퇴를 할 수 있다. 이런 경우에는, Successor의 id가 앞서 예상한 것과 같이 이상적일 수 없을 것이다. 예를 들어, Participant i의 j번째 Successor로 가정하는 $i+2^j$가 아직 시스템에 가입하지 않은 경우에는 id가 $i+2^j$보다 크면서 가장 작은 id를 가진 Participant를 Successor로 갖는다. 그림 3의 Participant 0, 2, 3의 사례가 그러하다. Successor에 해당하는 Participant 4가 아직 시스템에 가입하지 않아, Participant 중 4보다 크면서 가장 작은 id를 가진 5를 Successor로 지정한다[5].

임의의 Participant k에서 메시지가 발생하면, Successor List의 Participant들에게 메시지가 전달된다. 이 때, 메시지 내에 각 Successor가 담당해야 할 id 범위를 지정한다. 즉 j번째 Successor에게 $(k+2^j,\ k+2^{j+1}-1)$ 에 해당하는 Participant들에게 메시지를 전달해 달라는 요청을 메시지에 추가하여 전송한다. 이후로 이 범위를 'Broadcast Range'라고 부르기로 한다. 전송된 메시지를 수신한 j번째 Successor는 이 Broadcast Range 안에 해당하는 Successor들에게 메시지를 전달한다. 이 때, Participant k와 마찬가지로 각 Successor가 담당해야 할 Broadcast

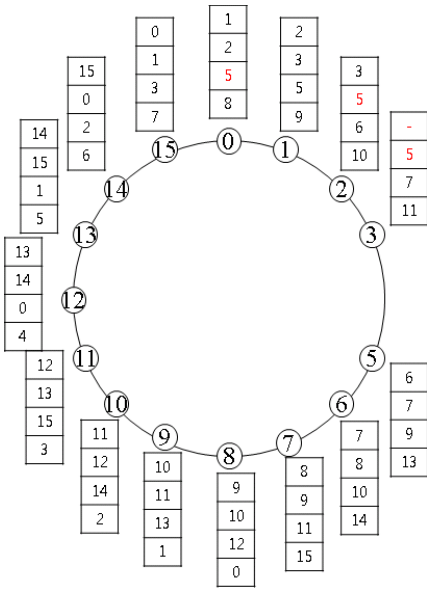


그림 3. Participant 4가 존재하지 않는 경우의 Successor List
Fig. 3. Successor list when participant 4 has not joined yet

Range를 지정한다. 이런 절차가 반복되면 그림 4에서 보이는 바와 같이, 시스템 내의 모든 Participant에게 해당 메시지가 전송된다[5]. 이후 본문에서는 이 기법을 DHTbroadcast라고 칭한다.

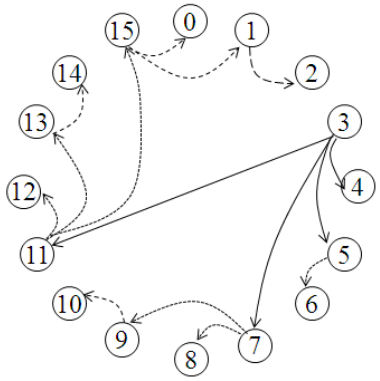


그림 4. Successor list를 활용한 메시지 확산 과정 [5]
Fig. 4. Message distribution process according to the successor list

그림 4는 ID가 3인 Participant가 전송하기 시작한 메시지가 시스템 내 전체 Participant로 전달되는 과정을 보여주고 있다. Participant 3의 Successor는 4, 5, 7, 11 이고, 각 Successor의 Broadcast Range는 4~4, 5~6, 7~10, 11~2가 된다. Participant 3에서 Successor 11에게 전달된 메시지는 Participant 11의 Successor인 12, 13, 15에게 전달된다. Successor 13에게 전달된 메시지는 다시 Successor 14와 15에게 전달되고, Successor 15는 이 메시지를 다시 Successor 1에게 전달하고, 이 메시지는 Successor1에서 Successor 2으로 전달된다. 이와 같은 방식으로 최소 1번 최대 4번의 메시지 전달로 네트워크에 존재하는 모든 Participant에게 전달된다. 그림 4의 메시지 확산 과정에서, 메시지가 한 홉 전달되는 것을 Edge로 표현하면, 메시지의 소스가 되는 Participant부터 나머지 Participant까지의 메시지 전달 과정이 그림 5와 같이 표현될 수 있다.

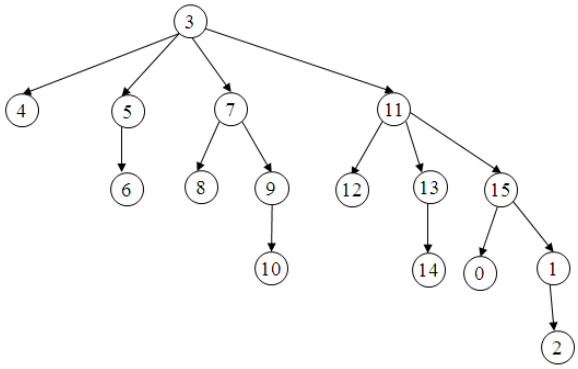


그림 5. 그림 4의 확산 과정을 표현한 Tree
Fig. 5. Message distribution tree corresponding to Fig.4

메시지 전송 부담을 고려한다면, 각 Participant의 입장에서 최대 $\log_2$maxID 개의 메시지를 전송하고 $\log_2$maxID 개의 ACK을 수신하면 되고, $\log_2$maxID 개의 메시지를 수신하고 $\log_2$maxID 개의 ACK을 전송하면 된다. 즉 전송되는 메시지의 개수만을 고려하면 각 Participant의 입장에서 기존의 DDS 표준에서 $O(e \times n)$의 복잡도를 갖는데 반해, 제안하는 기법은 $O(\log_2 \text{maxID})$의 복잡도를 갖게 되므로 n과 maxID가 비슷하다고 할 때, 본 논문에서 제안하는 기법이 그 복잡도를 크게 낮출 수 있다.

본 논문에서의 탐색 기법에서는 기존의 DHTbroadcast 기법[6]에 대해 다음의 두 가지 기능을 추가한다. 첫 번째, 메시지의 소스가 되는 Participant와의 거리를 계산하기 위해, 메시지 내에 전달 홉 수를 추가한다. 메시지의 소스는 전달 홉 수를 1로 기록하여 전달하고, 각 중간 Participant들은 자신이 수신한 메시지의 전달 홉 수에 1을 더하여 다음 Participant로 메시지를 전달한다. 이는 신규 Participant에게 축적된 MAP 정보를 전달할 주체를 결정하기 위한 기초 정보로 사용될 것이다. 전달 홉 수가 작은 Participant가 역할을 담당할수록 네트워크

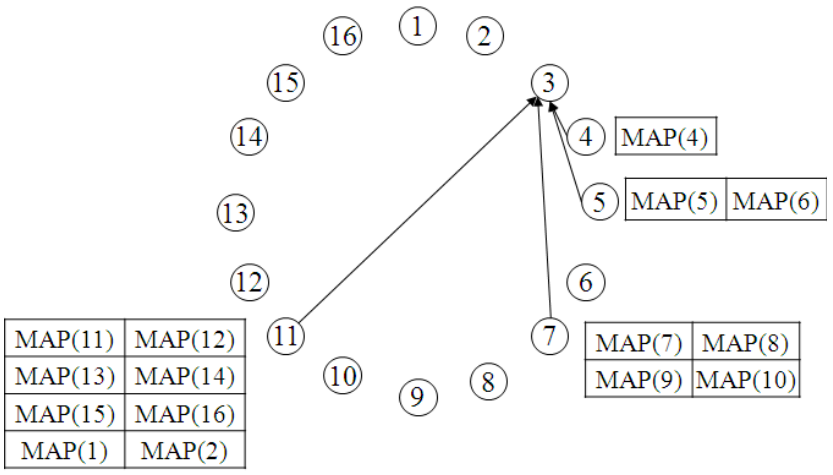


그림 6. 신규로 가입한 Participant에게 기존 Participant의 MAP 정보가 전달되는 과정 사례
Fig. 6. The example MAP distribution process of the existing participants to a new participant

내에 발생하는 메시지의 양을 줄일 수 있을 것이기 때문이다. 본 기법에서는 전달 홉 수가 1인 Participant가 역할을 담당하도록 한다.

두번째, DHTBroadcast의 메시지 전송 신뢰도를 높이기 위해 전송자와 수신자 간 ARQ 기법을 적용할 수 있다. 즉 하나의 Participant가 모든 Participant들이 자신을 발견했는지 확인하기 위해, 모든 Particpant들로부터 ACK을 수신하도록 할 수 있다. 그러나 이 경우, 시스템의 규모가 커질 때 이로 인한 메시지 발생량이나 재전송 부담 등이 크게 증가할 것이다. 따라서 DHTBroadcast의 각 단계에서의 송신자와 수신자 사이에 ARQ 기법이 적용되도록 한다. 즉 각 Participant는 자신의 Successor들에 대해서만 메시지가 오류없이 전송되었는지를 확인하면 된다. 이런 과정이 누적된다면 결국 모든 Participant들에게 메시지가 오류없이 전달될 수 있다.

3.2. 신규 참여 Participant 가입 절차

신규 Participant는 우선 자신의 id를 결정해야 한다. 각 Participant의 id는 DDS 표준에서 명시하는 것과 같이 IP 주소와 포트 번호 등을 조합하여 일정 범위의 숫자로 결정한다. 본 논문에서 제안하는 기법에서는 시스템마다 id를 부여하는 특정 서버가 존재한다고 가정한다. 이 초기화 서버는 전체 시스템을 관리하는 체계 안에서 구동하는 것이므로 시스템의 전체 구성 정보를 알고 있다면, id의 최대값인 maxID를 해당 서버가 알고 있어서 그 범위 안에서 임의로 id를 부여하는 것이 가능하다.

시스템에 가입하는 Participant들은 초기화 서버의 주소를 이미 알고 있으므로, JOIN 메시지를 전송하기 전에 우선 자신의 id를 수신하고 더불어 자신의 Successor List를 초기화 서버로부터 수신한다.

id가 k인 Participant는 시스템 내에 있는 모든 Participant에게 자신이 참여했다는 JOIN 메시지를 전달해야 한다. JOIN 메시지에는 Participant에 대한 정보와 Participant 내의 Endpoint들의 정보가 포함된다. 신규 Participant에서 발생한 JOIN 메시지는 DHTbroadcast 기법에 따라 시스템 내 모든 Participant들에게 k의 MAP 정보가 전달되게 된다. JOIN 메시지를 수신한 Participant 중에 신규로 가입한 Participant k가 자신의 Successor가 되어야 한다면 Successor List를 갱신한다.

신규 Participant의 MAP 정보가 시스템 내의 Participant들에게 전달되는 것과 동시에 시스템에 존재하는 모든 MAP 정보가 Participant k에게 전달되어야 한다. 전달 홉 수 필드가 1인 JOIN 메시지를 수신한 Participant들은 자신이 확보하고 있는 타 Participant의 MAP 정보를 JOIN_ACK 메시지에 포함하여 k에게 전달한다. 그 외의 Participant들은 JOIN메시지를 수신하였음을 알리기 위한 JOIN_ACK을 전송한다. 앞서 언급한 바와 같이 시스템 부팅 과정에서의 탐색 절차이므로 어느 한 Participant가 모든 타 Participant의 정보를 갖고 있을 수 없으므로 가능한 한 다수의 노드들로부터 정보를 획득하는 것이 필요하다.

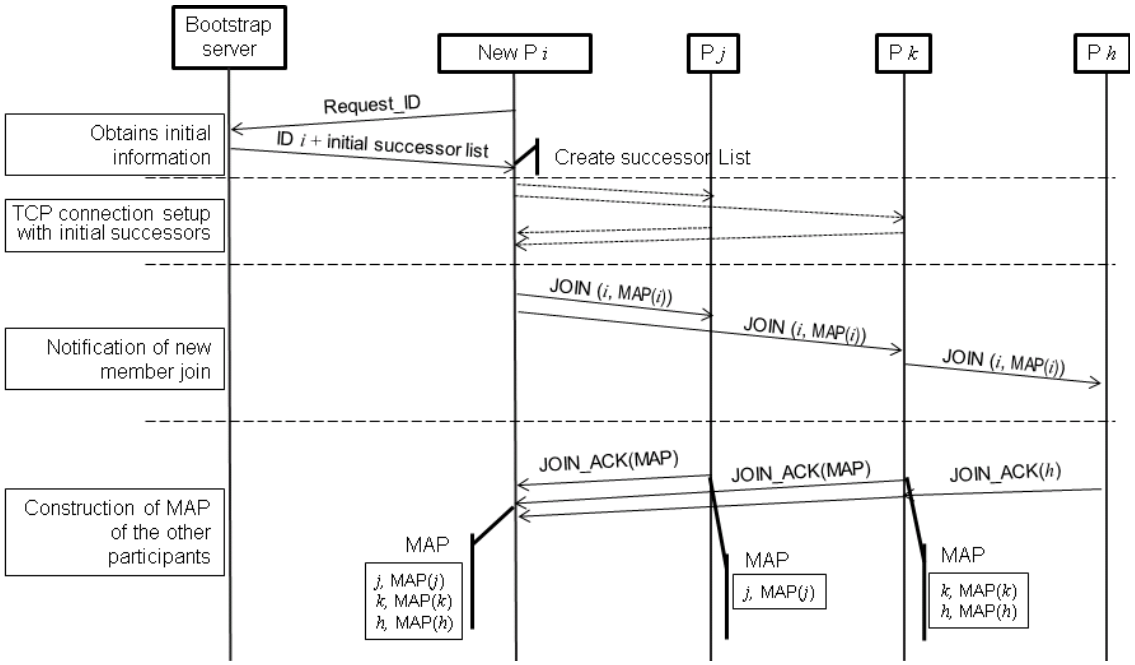


그림 7. 신규 Participant i가 시스템에 가입하는 절차
Fig. 7. Process for a new participant to join the system

ACK 메시지에 타 Participant의 MAP 정보가 전달되는 경우, id 공간상에서 JOIN 메시지의 소스 Participant와의 거리 d에 따라 포함되는 포괄하는 타 Participant의 범위가 달라진다. d가 1인 경우는 자신의 MAP 정보만 JOIN_ACK 메시지에 포함한다. d가 2인 경우는 자신과 id가 바로 인접한 Participant의 MAP 정보를, 그리고 d가 l인 경우는 자신의 id로부터 2^l-1의 범위 내의 Participant의 경우, 3에게 JOIN_ACK 메시지를 전달하는 Participant들이 포괄하는 타 Participant들의 id 범위를 보여주고 있다. Participant 4는 자신의 MAP 정보만 JOIN_ACK에 포함하여 전달하고, Participant 3과 가장 거리가 먼 Participant 11은 자신을 포함하여 Participant 11부터 16까지, 그리고 Participant 1과 2의 MAP 정보를 JOIN_ACK에 포함하여 Participant 3에게 전송한다.

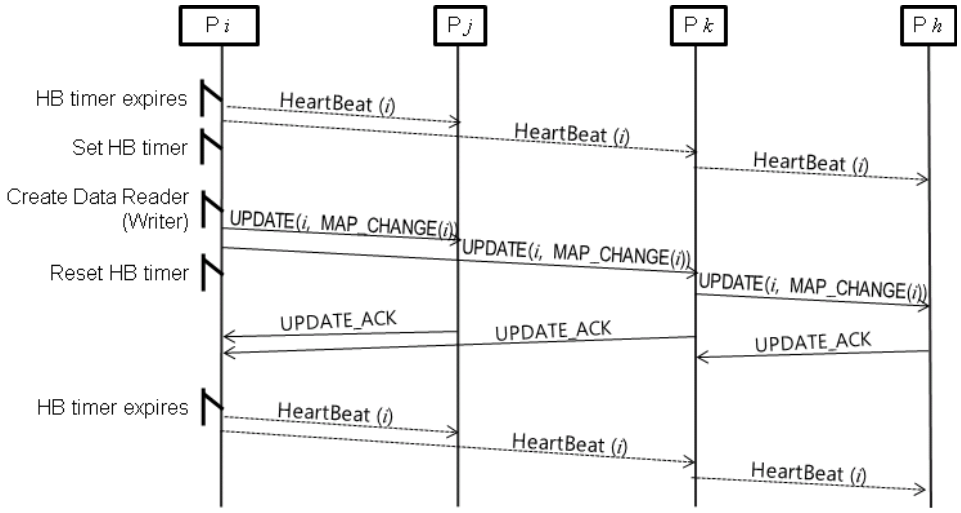


그림 8. Participant 정보 혹은 Participant 내 Endpoint 정보 변경을 공지하는 절차
Fig. 8. Endpoint or Participant change notification process

MAP 정보를JOIN_ACK 메시지에 포함한다.

그림 6은 Participant 3이 신규로 시스템에 가입한 그림 7은 Participant i가 새로 접속하는 경우의 처리 과정을 보여주고 있다. Participant i는

Bootstrap 서버로부터 Participant의 ID를 부여받고 Successor List를 설정한다. JOIN 메시지를 생성하여 자신의 Successor 인 Participant *j*와 *k*에게 TCP 연결을 설정하여 JOIN 메시지를 전달한다. Participant *j*는 *i*로부터 직접 JOIN 메시지를 수신하였으므로, 자신의 MAP 정보를 포함한 JOIN_ACK 메시지를 생성하여 Participant *i*로 전송한다. 반면 Participant *k*는 JOIN 메시지에 표시된 Broadcast Range내에 자신이 알고 있는 Participant인 *h*에게 JOIN 메시지를 전달한다. 그리고 *j*와 마찬가지로 자신이 확보하고 있는 MAP 정보를 모두 포괄하여 Participant *i*로 JOIN_ACK메시지를 전송한다. 메시지를 받은 Participant *i*는 필요한 경우, 자신의 Successor list를 갱신한다.

시스템 가입 절차를 마친 Participant는 자신이 동작중인 사실을 타 Participant들에게 알리기 위해 HeartBeat 메시지를 DHTbroadcast 기법을 사용하여 시스템 내 모든 Participant들에게 전달한다. 이 메시지의 목적이 단순하므로, 이 메시지에는 JOIN 메시지처럼 많은 정보를 포함할 필요없이 Particpant id 정도만 포함한다. HeartBeat 메시지의 크기가 크지는 않지만 다수의 HeartBeat 메시지가 동시에 발생하는 데이터 메시지 전달에 장애가 될 수 있으므로, HeartBeat 메시지 전달 주기는 일정 범위 안에서 각 Participant가 임의로 설정할 수 있게 한다.

3.3. 기존 MAP 정보 변경 절차

시스템 가입 절차를 완료한 Participant에서 Endpoint가 생성되거나 삭제되거나 변경이 일어나는 경우, DHTbroadcast 기법을 사용하여 시스템 내의 전체 Participant들에게 UPDATE 메시지를 전달하여, 자신의 변화를 알린다. UPDATE 메시지에는 기존의 MAP 정보 대비 변경된 사항만을 포함한다.

그림 8은 Participant *i*의 Endpoint가 새로 생성된 사례를 보여준다. 이때에는 UPDATE 메시지를 통해 Successor인 *j*와 *k*에게 UPDATE 메시지를 전송한다. Participant *k*는 자신의 Successor인 Participant *h*에게 UPDATE 메시지를 전달한다. 즉 UPDATE 메시지는 DHTbroadcast 기법에 따라 Successor Participant들을 통해 멀티 홉을 거쳐 시스템의 모든 Participant들에게 확산된다.

UPDATE 메시지가 HeartBeat 메시지와 마찬가지로 해당 Participant가 살아있다는 사실을 통지하는 효과가 있으므로, UPDATE 메시지를 전송하고 나면 HeartBeat 타이머가 초기화된다. 그림 8에서는 UPDATE 메시지 전송으로 초기화된 HeartBeat 타이머로 인해, 점선으로 예정되어 있던 HeartBeat 메시지 전송이 취소되고 좀 더 시간이 지난 후에 HeartBeat 메시지가 전송된 상황을 보여주고 있다.

3.4. 탈퇴 절차

시스템 참여를 종료하는 Participant는 Successor들에게 LEAVE 메시지를 전송한다. LEAVE 메시지는 DHTbroadcast 기법을 사용하여 시스템 내에 있는 모든 Participant들에게 확산된다.

그림 9는 Participant *i*가 시스템에서 탈퇴하는 과정을 보여준다. Participant *i*가 LEAVE 메시지를 생성하고 Successor Participant *j*와 *k*에게 전송한다. Participant *k*는 LEAVE 메시지를 자신의 Successor인 *h*에게 전달한다. Participant *i*는 자신의 Successor들인 *j*와 *k*로부터 LEAVE_ACK을 수신하면, 자신의 LEAVE 메시지가 시스템 내에 전체적으로 확산될 것으로 가정하고 시스템에서 탈퇴한다.

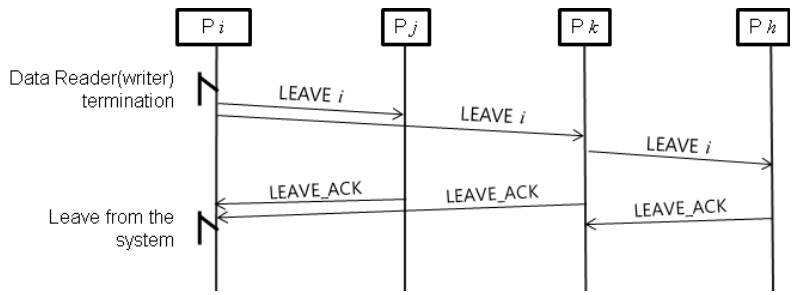


그림 9. 시스템에서 Participant가 탈퇴하는 과정
Fig. 9. Member leave process

Ⅳ. 성능평가

4.1. 시뮬레이션 환경

본 논문에서는 제안하는 기법의 성능을 평가하기 위해 QualNet ver. 5.2[10]을 사용하여 제안하는 기법을 시뮬레이션하였다. 기존의 DDS 표준에서 제시하는 탐색 기법을 SDP라 칭하고, 본 논문에서 제안하는 기법을 FDP라 칭한다.

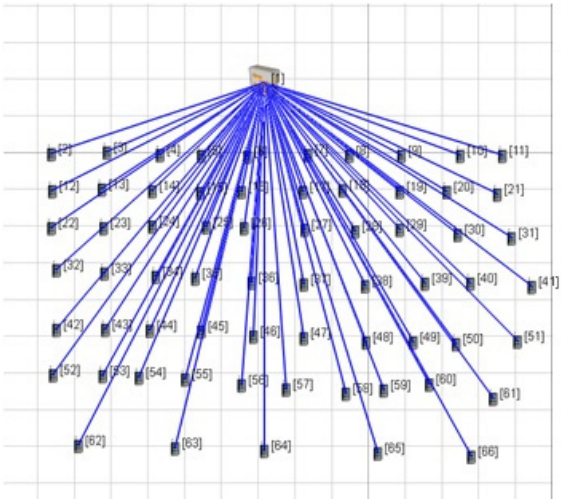


그림 10. 네트워크 구성
Fig. 10. Simulation network configuration

네트워크 토폴로지 구성은 하나의 스위치에 다수개의 노드가 유선으로 연결되어 있는 환경으로 구성하였다. 본 논문에서 고려하는 환경은 Participant와 Endpoint가 밀집한 상황을 가정하고 있기 때문이다. 함정의 경우, 하나의 라우터 하에 여러 개의 스위치를 배치하고, 각 단말들이 이 스위치 혹은 허브를 통해 연결되는 구조일 것이다. FDP의 경우 서버가 필요하기 때문에 노드를 하나 추가하여 Bootstrap 서버를 배치하였다. 링크 계층 프로토콜로는 IEEE 802.3을 사용하였고, 링크의 대역폭을 10Mbps, 100Mbps, 1Gbps, 10Gbps로 변화시켜 가면서 성능을 비교하였다. 그림 10은 64개의 Participant가 존재하는 네트워크 토폴로지를 보여주고 있으며, 표 2는 시뮬레이션 환경 변수 별 값을 보여주고 있다.

총 2가지 Case로 9가지의 시나리오에 대하여 시뮬레이션을 하였다. Case 1에서는 DDS 시스템 내의 각 Participant 별 Endpoint의 수를 고정시킨 상태에서 Participant 수를 8개에서 64개로 증가시키면서 성능을 분석하였다. Case 2에서는 64개의 Participant가 존재할 때, Endpoint의 수가 두 배씩 증가하는 환경을 5가지 시나리오로 시뮬레이션 하였다. Participant들 중에는 다수의 Topic을 생성하고 소모하는 경우도 있고, 그렇지 않은 경우가 있는 것을 가정하였다. 후자의 경우를 Level-1 Participant라 명명하고, 전자의 경우를 Level-2 Participant라 명명하였다.

표 2. 시뮬레이션 환경
Table 2. Simulation Environment

Parameter	Value
Link bandwidth	10Mbps, 100Mbps, 1Gbps, 10Gbps
Link layer protocol	IEEE 802.3
Network layer protocol	IPv4
Queue size at the switch	150Mbytes

4.2. SDP와 FDP 성능 비교

4.2.1. Participant 수에 변화에 따른 탐색 시간

각 Participant의 '탐색 시간'은 시스템 내의 해당 Participant가 발송한 개별 Endpoint_data 메시지가 나머지 Participant들에게 전달되는데 걸리는 시간의 최대값을 측정하였다. 탐색에 소요되는 시간을 계산해야 하고, 본 연구에서 추구하는 바가 메시지 전달 부담을 감소시키기 위한 것이므로, 메시지 관점에서의 소요 시간을 측정하였다. 그림 11은 각 Participant들의 탐색 시간 중 가장 큰 값을 표시한 것이다. 시스템 차원에서의 탐색 완료는 모든 Participant들의 시스템 내의 모든 Endpoint를 발견해야만 하는 것이므로, 최대 탐색 시간을 관찰하는 것이 타당하다.

그림 11은 표 3에 제시된 시나리오 별로 네트워크 대역폭을 달리하면서 관찰한 최대 탐색 시간을 나타낸다. 그래프의 X축을 Endpoint의 수로 나타냈는데, 이는 각 Participant의 Endpoint 수가 지정하여 결과적으로 Participant의 수 증가는 Endpoint의 수의 증가를 유도하기 위한 것이기 때문이다.

SDP는 Participant가 8개일 때 최대 탐색 시간은 7초이고, Participant가 64개일 때 최대 52초로 관측되었다. 이에 비해, FDP의 경우 Participant가 8개일 때 최대 탐색 시간은 0.5초이고, Participant가 64일 때에도 2.3초 정도이다. SDP의 경우, 실제 관측 결과를 보면 대부분의 Endpoint들이 수 초 이내에 발

표 3. 시나리오 별 Participant와 Endpoint 구성 내역
Table 3. Simulation scenarios with different Participants and Endpoints configuration

Scenario ID	Total number of Participants	Number of Participants per level		Number of endpoints per participant per level		Total number of Endpoints
		Level-1	Level-2	Level-1	Level-2	
Case 1-1	8	7	1	79	630	1,183
Case 1-2	16	14	2	79	630	2,366
Case 1-3	32	28	4	79	630	4,732
Case 1-4	64	56	8	79	630	9,464

견되는 반면, 몇 몇 Endpoint 정보에 대해 발견되는 데 걸리는 지연 시간이 길어져서 최대 탐색 시간이 길어진다. 이는 Endpoint_data가 UDP를 사용하여 전송되므로, 네트워크에서 손실되면 DDS 차원에서 주기적인 HeartBeat 메시지 전송에 의해서만 해당 Endpoint_data가 타 Participant들에게 전달되기 때문이다.

이에 반해, FDP는 TCP를 기반으로 하여 Successor들과 신뢰성 있는 메시지 전송을 지향하므로 네트워크에서 손실되더라도 SDP에 비해 훨씬 수월하게 오류를 복구할 수 있다. 다만 Participant의 수가 증가하게 되면, 그림 5가 보여주는 바와 같이 DHTbroadcast가 초래하는 Tree의 Depth가 커지므로 지연시간이 다소 증가한다.

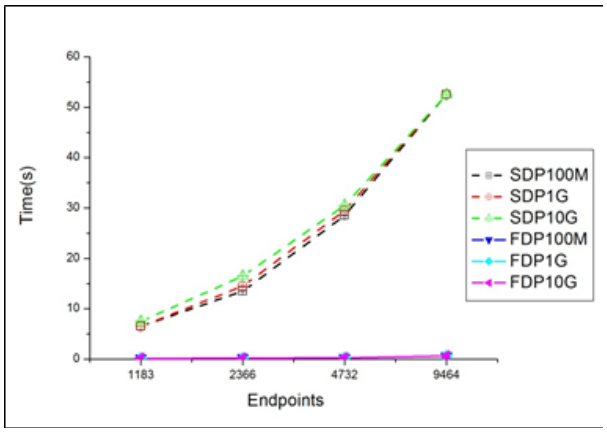


그림 11. 최대 탐색 시간
Fig. 11. Maximum discovery latency

그림 12는 각 Level에 속한 Participant의 Endpoint들이 타 Participant들에 의해 발견되는데 걸리는 시간의 평균값을 나타낸다. X 축은 시스템 내의 전체 노드 수를 의미한다.

SDP의 경우, 포함하고 있는 Endpoint의 수가 적은 Level-1 Participant의 경우, Endpoint들이 다 발견되는 데에 걸리는 시간이 약 5.5초인 반면, Endpoint의 수가 많은 Level-2 Participant의 경우 약 52.5초로 약 10배 정도의 차이가 났다. 이는 Level-2 Participant가 발생시키는 Endpoint_data 메시지의 수가 Level-1 Participant보다 10배 가량 많으므로, 해당 Participant의 Endpoint_data가 나머지 모든 Participant들에 의해 발견되는 데에 걸리는 시간이 Level-1 Participant보다 10배 정도 차이가 발생하는 것이다. 특히 Level-2 노드의 수가 증가할수록 탐색 시간이 늘어나는 비율이 더 커지는 것으로 관찰되었다.

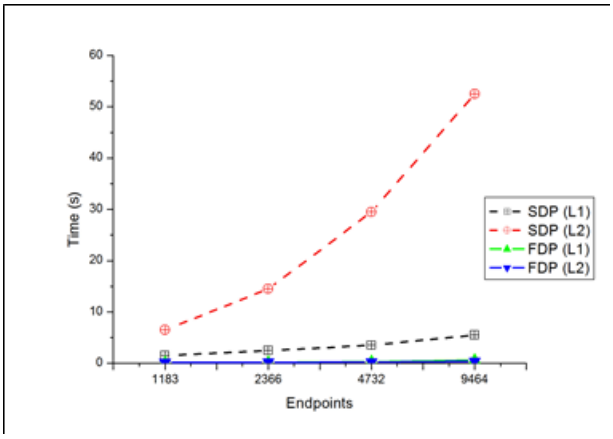


그림 12. Level별 최대 발견 시간
Fig. 12. Maximum discovery latency per level

그림 13은 메시지 수신률을 보여주고 있다. 각 Participant가 탐색 과정에서 전송한 메시지의 양 대비 타 Participant에게 수신된 메시지의 양의 비율을 측정한 값이다. SDP의 경우 송신한 메시지에는 PDP 단계의 Participant_data 메시지, Heartbeat 메시지, EDP 단계의 Endpoint_data, ACK, Heartbeat과 HeartBeat_ACK/ NACK 등이 포함된다. FDP는 Request_ID, JOIN, JOIN_ACK 등이 포함된다. SDP와 FDP가 전송한 메시지 양은 Participant 수가 많을수록 차이가 많이 나는데 Participant의 수가 64일

표 4. Participant Level에 따른 설정
Table 4. Simulation environment with different participant configuration for each level

Scenario ID	Total number of Endpoints	Total number of Participants	Number of Endpoints per Participant		Endpoint ratio of Level-1 Participant
			Level-1	Level-2	
Case 2-1	6400	64	18~19	604~605	10%
Case 2-2	6400	64	53~54	598~600	30%
Case 2-3	6400	64	83~84	587~588	50%
Case 2-4	6400	64	111~112	564	70%
Case 2-5	6400	64	136	470	90%

경우 SDP가 FDP의 약 26배 많이 전송한다. 이는 SDP의 경우, DDS 차원에서의 Endpoing_dat의 재전송으로 인해 메시지의 양이 늘어나 전체 트래픽 양이 증가했기 때문으로 분석할 수 있다. 반면에 FDP의 경우 정보를 한 번에 묶어서 전송하고 TCP로 신뢰성 있는 메시지 전송을 지원하므로 DDS 차원의 재전송 메시지의 양이 줄어들었기 때문에 성능 차이를 보이는 것으로 분석된다.

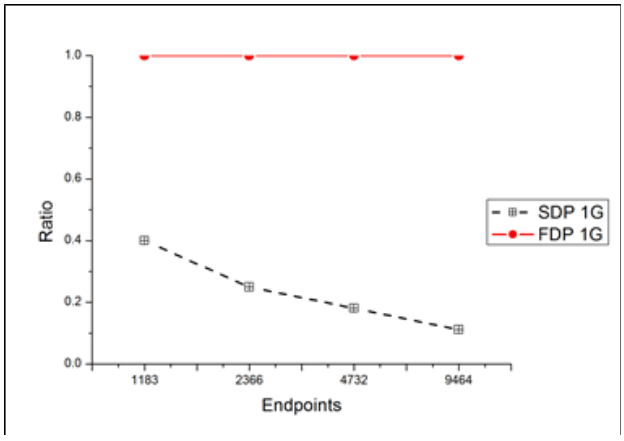


그림 13. 메시지 수신률
Fig. 13. Message reception ratio

4.2.2. Endpoint 비율에 따른 탐색 시간

Participant의 수를 64로 고정시키고 전체 Endpoint의 수를 6400으로 고정시킨 상태에서, Level-1 Participant들의 Endpoint의 수를 증가시키면서 탐색시간, 메시지 수신율을 측정하였다.

그림 14는 표 4에 제시된 시나리오 별로 관측된 최대 탐색 시간을 그래프로 나타낸 것이다. SDP의 경우는 Level-1에 속한 Endpoint들의 수가 증가할수록, 즉 Endpoint가 밀집된 Participant의 수가 줄어들수록, 최대 탐색 시간이 감소하는 것을 관찰할 수 있다. 링크 대역폭이 100Mbps일 때, Level-1과 Level-2 Participant에 속한 Endpoint의 비율이 10:90인 경우 약 51.5초로 가장 오래 걸리고, Endpoint의 비율이 90:10인 경우 약 36.52초로 가장 적게 걸린다.

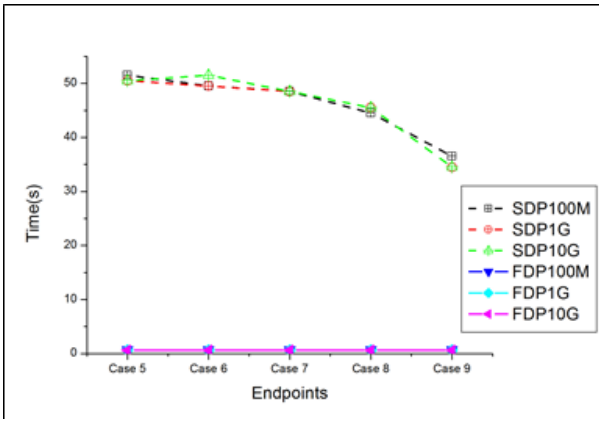


그림 14. Endpoint 비율 변화에 따른 최대 탐색 시간 변화
Fig. 14. Maximum discovery latency change according the endpoint ratio per participant level

FDP의 경우 Participant의 Endpoint 밀집도가 큰 영향을 주지 않는 것으로 나타났다. 개별 Participant 내에 Endpoint의 수가 늘어날수록 MAP 정보의 크기가 커지므로 탐색 시간을 크게 증가시킬 것으로 예상되지만, DHTbroadcast 기법으로 인해, 각 Participant 별로 전송해야 하는 대상이 적고 Successor들 간의 TCP 연결로 오히려 탐색 시간이 감소하는 효과를 보이고 있다. SDP와 FDP의 최대 탐색시간 Ratio는 54.75~81.74에 이른다. 본 연구에서 제안하는 FDP가 본 시뮬레이션에서 제안하는 네트워크 구조에 대해서는 SDP에 비해 확연하게 탐색 완료 시간을 줄일 수 있다.

그림 15는 탐색 과정에서의 메시지 수신률, 즉 송신한 메시지 대비 수신한 메시지의 비율을 보여주고 있다. FDP는 100%의 메시지 수신률을 보이고 있는

반면, SDP는 Level-1 Participant의 Endpoint 비율이 증가하는 것과 비례하여 수신률이 개선되는 것을 관찰할 수 있다. Level-1 Participant의 Endpoint의 비율이 10%인 경우 메시지 수신률 7.9%로 가장 낮고, 90%일 때 22.6%로 메시지 수신률이 가장 높게 나타났다.

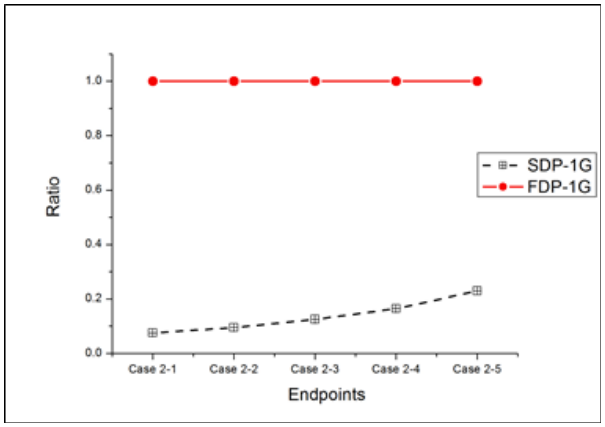


그림 15. Endpoint 비율 변화에 따른 메시지 수신률
Fig. 15 Message reception ratio change according to the endpoint ratio per participant level

이 실험에서 관찰된 결과를 해석하면 다음과 같다. 전체 Endpoint의 수가 동일하므로, 각 시나리오마다 발생하는 Endpoint_data 메시지의 개수는 동일하다. 그럼에도 불구하고, 개별 Participant가 전송하는 Endpoint_data 메시지의 수가 작을수록 해당 Participant의 Endpoint들이 모두 발견되는데 걸리는 시간이 감소하므로 전체적으로 탐색 완료에 필요한 소요 시간이 줄어든다는 것이다.

4.3. FDP 성능분석

4.3.1. 다중 홉 전송에 의한 효과 분석

SDP에서는 탐색 과정에서 개별 Participant들이 직접 메시지를 교환하지만, FDP에서는 Successor들을 활용하여 메시지 송수신 대상을 제한하였다. 그 결과 모든 Participant에게 메시지를 보내기 위해서는 여러 Participant들을 거쳐서 전송해야 한다. 그림 16은 FDP에서 탐색 과정에서 전송되는 메시지들이 최대 전달된 범위를 그래프로 표현하였다. Participant가 64일 때, 최대 6 홉을 거쳐 메시지가 전달된다. 이로 인해, 메시지 전달 지연 시간이 길어질 수 있으나, 본 논문에서 제안한 실험 환경에서는 이로 인해 지연 시간이 증가하는 현상은 미비하였다. 그리고 그림 16에서 볼 수 있는 것처럼 3 홉까지의 메시지 전달이 가장 빈번하게 발생하므로 다중 홉으로 인한 성능 저하는 크지 않다고 예상할 수 있다.

제안하는 기법에서 Participant와 Successor들 사이에 TCP를 활용하여 메시지를 전송하도록 하였으므로, 매 메시지마다 TCP 연결을 새로 설정한다면 TCP 연결 설정 과정에서의 지연 시간이 전체 메시지 전달 지연 시간을 증가시키는 요소가 될 수 있다. 그러나 앞서 3.2에서 기술한 바와 같이, 시스템 가입 초기에 Successor들과 TCP 연결을 설정한 이후로, 탈퇴까지 계속 사용하므로, TCP 연결 설정 부담이 전체 시스템의 성능에 끼치는 영향은 무시할 수 있다.

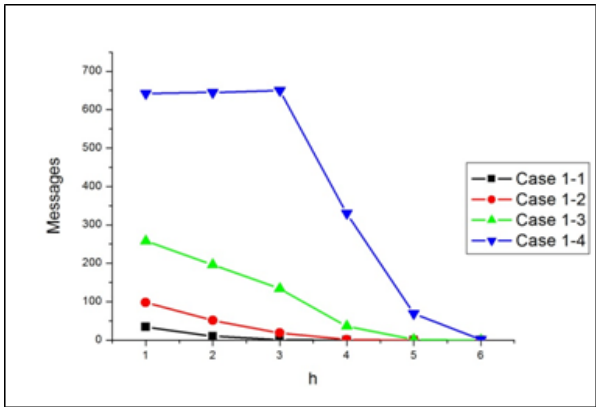
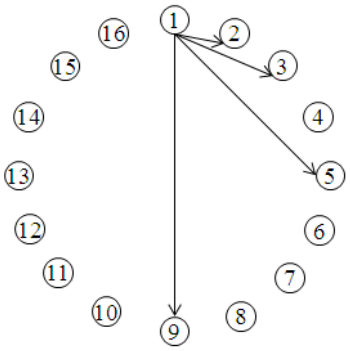
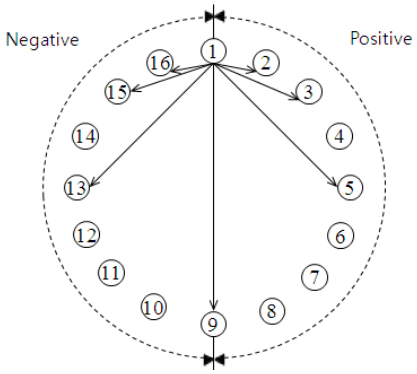


그림 16. 메시지를 전달하기 위한 Hop 수에 따른 전달된 메시지 수
Fig. 16. Number of messages sent, to distribute a single message according to the hop counts

본 실험에서 TCP 연결 활용으로 인한 다음의 문제 상황을 발견하고 개선 방안을 적용하였다. Participant들이 일순간에 시스템에 가입하는 것이 아니고 짧더라도 시간 간격을 두고 점진적으로 가입을 하게 된다. 새 Participant 가입으로 인해 기존 Participant의 Successor가 변경되어야 하는 경우가 발생한다. 그림 3에 있는 Participant 4가 시스템에 가입함으로 인해서 Participant 1, 2, 3의 Successor가 변경되는 경우를 생각할 수 있다. Successor가 바뀌게 되면 Participant는 기존에 Successor를 대행했던 participant, 예를 들어, 그림 3의 Participant 5와의 Connection을 끊고, 새로운 Successor인 Participant 4와 TCP 연결을 생성하게 된다. 이때 TCP의 3-way Handshaking으로 인해 약간의 지연이 발생한다. 이러한 지연 시간은 매우 짧기 때문에 대부분의 경우 무시될 수 있다. 하지만 시스템의 노드의 수가 증가하면서 동시에 다수의 Participant가 가



(a) Participants in one hop range in case of FDP



(b) Participants in one hop range in case of BiFDP

그림 17. FDP와 BiFDP 비교
Fig. 17. Comparision between FDP and BiFDP

입하게 되면 새로운 Successor와 TCP 연결이 만들어지는 동안 발송되는 JOIN 메시지를 손실하게 되는 경우가 발생한다. 따라서 새 Successor와의 TCP 연결을 완성하기까지는 기존의 Successor와의 연결을 끊지 말고 유지해야 한다.

4.3.2. 최대 메시지 크기

FDP는 각 Participant가 가지는 Endpoint의 개수와 Successor의 수에 따라 JOIN_ACK 메시지의 크기가 달라진다. JOIN_ACK의 크기를 일반화하여 수식으로 표현하면 다음과 같다.

Participant k의 Participant data의 크기를 L_P라 표시하고, 하나의 Endpoint data의 크기를 L_E라고 표시하자. Participant k에 존재하는 Endpoint의 수를 $N_E(k)$라 표시한다. Participant가 시스템에 가입할 시에 broadcast하는 JOIN 메시지는 Participant data와 Endpoint data를 포함한다. 따라서 JOIN 메시지의 MAP 정보의 크기는 $L_{JOIN} = L_P + L_E \times N_E(k)$ 라고 표현할 수 있다.

JOIN 메시지를 받은 Successor가 발생하는 JOIN_ACK 메시지는 broadcast range 내에 존재하는 모든 Participant의 MAP 정보를 포함하여야 한다. 따라서 broadcast range가 m인 Successor가 전송하는 JOIN_ACK 메시지의 크기는 식 (1)과 같다.

$$L_{JOINACK}(m) = L_P \times m + L_E \times \sum_{j=1}^{m} N_E(j) \quad (1)$$

m의 최대값은 $\log_2$maxID로, JOIN_ACK 메시지의 크기는 최대 $n/2$에 달할 수 있다. 이러한 메시지 전달 부담을 감소시키고 메시지 확산 속도를 향상시키기 위한 방안으로 4.3에서 양방향 이웃 노드를 활용하는 방안을 제시한다.

4.4. 양방향 이웃 노드를 활용한 FDP 성능 개선

3장에서 기술한 FDP는 Successor를 활용해서 MAP 정보를 점진적으로 확산하고 수집한다. 시스템 내에 가입된 Participant의 수가 커지면서 Successor의 수가 커지게 되면, MAP 정보를 수집하는데 지연시간이 길어지게 되고, 이로 인해 각 Participant들은 순간적으로 서로 다른 MAP 정보를 가질 수 있다. 이런 지연 시간을 줄이기 위해, 그림 17과 같이 Successor를 확대하고 Positive Successor와 Negative Successor로 구분한다. Positive Successor가 id가 증가하는 방향으로 정의되는 Successor라면, Negative Successor는 id를 감소시키면서 Successor를 정의한다. 이 기법은 양방향으로 Successor를 정의하게 되므로 Bidirectional FDP (BiFDP)라고 부른다.

같은 수의 Participant가 있는 시스템에서 BiFDP의 최대 홉 수는 FDP의 최대 홉 수보다 하나 적게 되고, 한 번에 JOIN_ACK에 포함해서 전달해야 하는 MAP 정보의 양도 1/2로 줄어든다. 그러나 한 Participant가 유지해야 하는 TCP 연결의 수가 증가하게 되고, 이에 따라 전송하는 메시지의 수가 증가하게 되므로, 링크 상에서 전송되는 정보의 양은 동일하다.

그림 17에서 Participant 9는 BiFDP를 적용하더라도 Participant 1의 JOIN 메시지를 수신하는데 필요한 홉 수는 동일하다. 그러나 Participant 16이 Participant 1의 JOIN 메시지를 수신하기 위해서, FDP는 4 홉의 메시지 전달이 필요하지만 BiFDP는 1번의 전달로 JOIN 메시지를 수신할 수 있다.

V. 결 론

본 논문에서는 Publish/Subscribe 기반 대표적인 미들웨어인 DDS의 시스템 부팅 시 서비스 탐색 단계의 긴 지연 시간 문제를 개선하기 위한 방안을 제시하였다. 본 논문에서 고려하는 서비스 환경은 좁은 지역에 다수의 밀집된 장치들과 서비스들이 DDS를 사용하여 데이터를 교환하는 환경이다. 기존의 DDS 표준은 Participant 탐색 단계와 Endpoint 탐색 단계로 분리되어 있으며, 두 Participant 간에 UDP를 사용하여 개별 Endpoint 정보에 대해 메시지를 송수신하는 과정에서 발생하는 다수의 메시지 손실로 탐색 시간이 길어진다.

본 논문에서 제안하는 기법 FDP는 시스템 부팅 초기에 Participant 탐색과 Endpoint 탐색을 통합한다. 그리고 DHT 기법의 하나인 Chord의 Successor 개념을 도입하여, 개별 Participant가 탐색에 필요한 메시지를 교환해야 할 대상을 줄일 수 있다. 또한 DHTbroadcast 개념을 도입하여 TCP를 사용해서 점진적으로 메시지를 전체 Participant들에게 확산시킬 수 있다. 시뮬레이션을 통한 성능 평가에서 단말과 서비스들이 밀집된 환경에서 FDP가 기존의 DDS 표준에 비해 확연하게 탐색 완료 시간을 단축시키는 것을 확인할 수 있었다. 본 논문의 목적이 시스템 부팅 단계에서의 긴 지연 시간을 감축시키는 것이므로 본 논문에서 제안하는 기법은 시스템 부팅 단계에서 적용하고 시스템에 구동되고 있는 상태에서는 DDS 표준 탐색 기법을 적용할 수 있다.

추후로 두 가지 단계로 진행할 것이다. 첫째, 기존 연구 결과를 보강하기 위한 활동을 진행할 것이다. 본 연구는 특수한 적용 환경에서의 문제 상황을 극복하기 위한 방안으로서 제시되었으므로, 일반적인 환경으로의 연구로 확장할 것이다. 또한, 본 논문에서 중점적으로 고려한 환경에서 충돌로 인한 손실과 부담이 얼마인지 구체적인 결과를 얻기 위한 심도 있는 실험을 수행하도록 할 것이다. 그리고, Participant 혹은 Endpoint들이 시스템에 가입하고 탈퇴하는 과정에서 발생하는 탐색 메시지 처리 부담 등을 분석할 것이고, TCP를 시스템 가입부터 탈퇴까지 유지하는 것에 따른 시스템 성능 영향에 대해서 분석할 것이다. 그리고, 마지막으로 제시한 BiFDP에 대한 성능 분석을 진행할 것이다. 둘째, 본 연구를 확장하기 위한 연구를 진행할 것이다. 가입 단말에게 ID를 부여하는 기능을 중앙의 서버가 아닌 분산된 방법으로 단말들이 자율적으로 ID를 충돌없이 결정하는 방법을 강구할 것이다. 그리고, 서로 다른 도메인에 속한 Participant들 간의 탐색 시간을 단축하기 위한 방안으로 본 연구의 적용 범위를 확장할 것이다.

References

[1] Object Management Group, *Data Distribution Service for real-time systems specification*, Ver. 1.2, 2007.

[2] RTI, *RTI DDS*, retrieved Mar, 1, 2011, from http://www.rti.com.

[3] Object Management Group, *The real-time publish-subscribe wire protocol DDS interoperability wire protocol specification*, Ver. 2.1, 2009.

[4] S. Chae, S. Ahn, K. Kang, J. Kim, S. Lee, and W. Kim, "Fast discovery scheme using DHT-like overlay network for a large-scale DDS," *Commun. Comput. Inform. Sci.*, vol. 256, pp. 128-137, Dec. 2011.

[5] S. Ahn, N. Kim, K. Kang, and Y. Cho, "Fast discovery scheme using overlay network for DDS device," in *Proc. Korea Inform. Commun. Soc. (KICS) Conf.*, pp. 329-330, Seoul, Korea, Nov. 2012.

[6] S. El-Ansary, L. Alima, P. Brand, and S. Haridi, "Efficient broadcast in structured P2P networks," in *Proc. Int. Workshop Peer-To-Peer Syst. (IPTPS)*, pp. 304-314, Berkeley, U.S.A., Feb. 2003.

[7] J. S. Monedero, "A DDS discovery protocol based on Bloom filters," M.S. Thesis, Dept. Inform. Sci. Telecommun. Eng., Universidad de Granada, Sep. 2009.

[8] M. Christensen, K. Kimball, and F. Solensky, *Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches*, IETF RFC 4541, May 2006.

[9] I. Stoica, R. Morris, D.R. Karger, M.F. Kaashoek, and H. Balakrishinan, "Chord: a scalable peer-to-peer lookup service for internet applications," in Proc. ACM SIGCOMM, pp. 149-160, San Diego, U.S.A., Aug. 2001.

[10] Scalable Network Technologies, *The Simulator: QualNet*, retrieved Mar., 1, 2011, from http://www.qualnet.com.

안 시 내 (Si-Nae Ahn)

2011년 2월 아주대학교 정보컴퓨터공학부 학사
2013년 2월 아주대학교 대학원 컴퓨터공학과 석사
<관심분야> neighbor discovery, 멀티캐스트, publish/subscribe system

강 경 란 (Kyungran Kang)

1994년 2월 KAIST 석사
1999년 2월 KAIST 박사
2004년 3월~현재 아주대학교 부교수
<관심분야> 멀티캐스트, 이동 네트워크, 전술 통신, publish/subscrib system

조 영 종 (Young-Jong Cho)

1985년 2월 KAIST 석사
1990년 2월 KAIST 박사
1996년 3월~현재 아주대학교 교수
<관심분야> 멀티캐스트, 무선 네트워크, 트래픽 모델링

김 노 원 (Nowon Kim)

2006년 2월 동명대학교 컴퓨터공학과 학사
2008년 3월~현재 아주대학교 컴퓨터공학과 석박사통합과정
<관심분야> 멀티캐스트, 무선 이동 네트워크, 전술통신, P2P