DDS 미들웨어의 상호운용성 제공을 위한 표준 디스커버리 프로토콜

안성우 · 최종우 · 최윤석
삼성탈레스 해양/시스템연구소

Standard Discovery Protocol for Supporting Interoperability between DDS Middlewares

Sung-woo Ahn · Jong-woo Choi · Yoon-suk Choi
Naval/System R&D Center, Samsung Thales Co., Ltd.
E-mail : {sungwoo.ahn · jongwoos.choi · yoonsuk007.choi}@samsung.com

요 약

최근 분산 환경에서 실시간 데이터 교환에 대한 요구가 증가하면서 발간/구독(publish/subscribe) 기반의 데이터 중심 통신 미들웨어인 DDS (Data Distribution Service)에 대한 관심 또한 증대되고 있다. 현재 다수의 벤더 및 연구단체에서 DDS 미들웨어를 제공하고 있으나 이들 간에 메시지 교환 포맷, 디스커버리 방식 등의 차이로 인해 상호연동이 제대로 되지 않는 문제가 있다. 이에 따라, OMG에서는 네트워크를 통한 메시지 교환을 위한 표준명세인 RTPS (Real-Time Publish-Subscribe)를 정의함으로써 서로 다른 DDS 간의 상호운용성을 제공하고자 하고 있다. 본 논문에서는 DDS의 핵심 기능인 디스커버리의 상호운용성을 위해 RTPS에서 정의하고 있는 SDP (Simple Discovery Protocol)에 대해서 분석하며 SDP의 구현을 위한 효율적인 설계방안을 제공한다.

ABSTRACT

In recent years, the user interest has increased for DDS (Data Distribution Service) which is a data centric middleware based on publish-subscribe communication as the demands for real-time data exchange in distributed systems have been growing rapidly. To reflect these needs, many vendors and research groups provide their DDS middleware. However, there has been a problem with interoperability between DDS middlewares because of a lack of common communication rules such as the message exchange and the discovery manner. For this reason, OMG defines RTPS (Real-Time Publish-Subscribe) specification which is the standard network protocol used to exchange data between different implementations of DDS. In this paper, we analyze and design the SDP (Simple Discovery Protocol) of RTPS which enable DDS middleware to provide interoperable discovery mechanism.

키워드

DDS, 발간/구독, 디스커버리, SPDP, SEDP

Ⅰ. 서 론

최근 이슈가 되고 있는 온라인 비디오, 소셜 네트워크 서비스와 같이 유비쿼터스 환경을 기반으로 한 분산 네트워크에서는 임베디드 시스템, 모바일 기기, 센서 등 다양한 장치들로부터 대량의 데이터가 끊임없이 생성, 배포되는 특징이 있다. 이러한 데이터는 네트워크 참여자 간에 실시

간으로 공유가 요구되며 시간의 경과에 따라 데이터 전송에 참여하는 참여자들의 동적인 변경을 수용할 수 있는 데이터 중심 통신(Data Centric Communication) 기술을 요구하고 있다. 데이터 중심 통신을 위한 많은 기술들이 존재하고 있으나 그 중 발간/구독(publish/subscribe) 기술이 많은 응용분야에 핵심 기술로 자리잡고 있다[1,2].

OMG (Object Management Group)에서는 분산 응용프로그램 간의 데이터 중심 통신을 지원하기 위해 DDS (Data Distribution Service) 미들웨어에 대한 표준을 제공하고 있다. DDS 미들웨어는 중앙 서버없이 단순한 발간/구독 방식의 통신을 사용함으로써 분산 환경에서 실시간 데이터 배포를 효과적으로 할 수 있다. DDS 표준은 크게 DCPS (Data-Centric Publish Subscribe) [3]와 RTPS (Real-Time Publish-Subscribe) [4]의 두 개의 계층 구조로 정의되어 있다. DCPS는 데이터 모델 정의를 통해 DDS 응용으로 표준 인터페이스와 데이터 교환 최적화를 위한 QoS 설정 기능을 제공한다. RTPS는 네트워크를 통한 DDS 데이터 패킷 전송 포맷에 대한 규격을 정의하고 있다.

서로 다른 벤더에서 제공하는 DDS 미들웨어 간에 상호연동을 위해서는 RTPS 계층을 제공하는 것이 필수적이다. RTPS는 DDS 미들웨어 간 연동을 위해 첫째, 네트워크를 통한 메시지 교환 표준 포맷을 정의하며, 둘째, 통신에 참여하는 개체(Entity)들 간에 서로의 QoS를 확인하고 통신을 맺을지에 대한 판단기준이 되는 표준 디스커버리 프로토콜을 정의한다. 만약 RTPS 계층을 사용하지 않는다면 각 미들웨어마다 독자적으로 정의한 메시지 교환 포맷 및 디스커버리 방식으로 통신을 하게 됨으로써 동일 벤더의 DDS가 아니면 데이터 교환이 불가능한 문제가 발생한다.

본 논문에서는 DDS 미들웨어 간 상호연동의 요구사항 중 RTPS 계층에서 제공해야하는 표준 디스커버리 프로토콜인 SDP (Simple Discovery Protocol)에 대해서 다룬다. RTPS에서 SDP 기능의 효율적인 구현을 위해 본 논문에서는 SDP를 분석하여 동작 메커니즘을 도출하고 이를 기반으로 한 프로토콜 설계 방안을 제시한다.

Ⅱ. RTPS 디스커버리 프로토콜

DDS 미들웨어를 통해서 교환되는 데이터는 토픽(topic)으로 정의되며, 토픽을 교환하기 위해서는 네트워크에 참여하는 개체인 참여자(Participant)와 데이터 발간/구독의 주체가 되는 단말개체(Endpoint)가 서로 관심을 가지는 토픽을 알고 있어야 한다. 이를 위해서 RTPS 표준명세에서는 DDS 미들웨어에서 제공해야 할 공통 디스커버리 프로토콜인 SDP를 정의하고 있다. SDP는 다시 참여자 검색을 위한 SPDP (Simple Participant Discovery Protocol)과 참여자에 속한 단말개체를 검색하기 위한 SEDP (Simple Endpoint Discovery Protocol)로 나누어진다. 모든 DDS 미들웨어는 디스커버리 성능의 최적화를 위해 별도의 프로토콜을 제공할 수 있지만 미들웨어 간 연동을 위해서는 SPDP와 SEDP를 반드시 제공해야 한다 [4].

가. SPDP

단말개체 간에 서로 토픽을 교환하기 위해서는 먼저 단말개체가 속해있는 참여자 간에 정보를 교환하는 것이 필요하다. 이를 위해 SPDP 단계에서는 SPDPdiscoveredParticipantData 구조체를 이용하여 참여자의 정보를 표현하며 참여자가 생성된 후 주기적으로 이 데이터를 네트워크에 멀티캐스트로 전송한다. 다른 참여자는 상대편의 참여자 정보가 수신이 되면 미리 정의된 내부 저장소에 저장한다. SPDPdiscoveredParticipantData의 송수신은 이를 위해 미리 정의된 단말개체인 SPDPbuiltinParticipantWriter와 SPDPbuiltinParticipantReader가 담당한다.

나. SEDP

참여자 간에 참여자 정보가 교환되고 나면 참여자 내부에 데이터를 교환하기 위해 필요한 단말개체의 정보를 교환해야 한다. 이때에도 SPDP와 비슷한 방법으로 토픽 송수신을 위한 Writer, Reader에 대해 미리 정의된 구조체인 DiscoveredWriterData, DiscoveredReaderData를 전송하며 수신측 참여자는 구조체 정보를 내부 저장소에 저장한다. 각 구조체는 단말개체에 대한 위치, 관련 토픽, QoS 등의 정보를 포함하고 있다. 이후 각 단말개체 간의 데이터 통신을 위해서 수신된 구조체에서 토픽 및 QoS를 검사하며 매칭되는 자신의 Writer, Reader 단말개체에 각각 ReaderProxy, WriterProxy를 연결해준다.

이와 같이 SPDP와 SEDP는 참여자와 참여자에 속해있는 단말개체가 생성, 변경, 삭제될 때마다 해당 정보를 참여자끼리 교환할 수 있도록 한다. 이를 통해 단말개체 간에 메시지를 올바르게 발간/구독을 할 수 있도록 한다.

Ⅲ. SDP 디스커버리 모듈 설계

가. 클래스 다이어그램

앞서 살펴보았듯이 SDP는 SPDP와 SEDP의 두 단계의 디스커버리 과정을 포함하고 있다. SDP 디스커버리에 참여하는 객체와 각 객체가 담당하는 작업 단위는 그림 1과 같이 정의될 수 있다. 디스커버리 작업은 Participant, SPDPManager, SEDPManager 객체가 각각 담당하며 각 단계의 작업은 II장에서 이미 설명하였다. 그림 1에서의 Writer, Reader는 SPDPManager, SEDPManager의 미리 정의된 단말개체로서도 사용되며 디스커버리가 완료된 후 토픽 정보를 교환하기 위한 단

말개체로서도 사용된다.

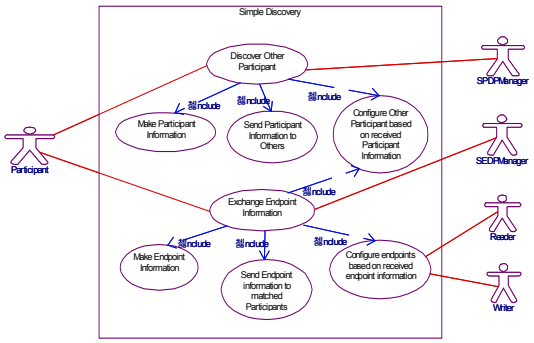

그림 1 SDP를 위한 객체와 단위작업 정의

그림 1에서 정의된 객체를 기반으로 구현을 하기 위해서 클래스는 그림 2와 같이 구성할 수 있다.

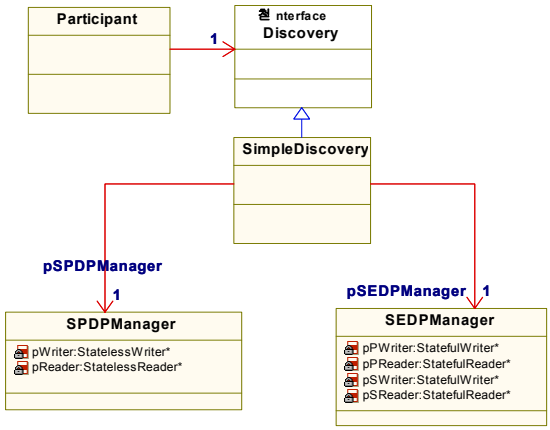

그림 2 SDP 모듈 클래스 다이어그램

단말개체의 생성 및 관리는 모두 참여자를 통해서 이루어지므로 Participant 클래스가 디스커버리 모듈을 포함한다. 그리고 RTPS 표준명세에서 언급하고 있듯이 RTPS는 향후 새로운 디스커버리 기능의 추가가 가능해야 한다. 이를 위해서 Discovery 인터페이스를 SimpleDiscovery 클래스가 상속하여 SDP를 구현하도록 한다. SPDPManager와 SEDPManager는 각각 SPDP, SEDP 디스커버리를 담당하는 모듈로 SimpleDiscovery에서 변수로 등록되어 있으며 각 디스커버리 단계에서 수행해야 할 작업은 각 매니저를 통해서 하도록 한다. 마지막으로 SPDPManager, SEDPManager는 미리 정의된 Writer, Reader 단말 개체를 등록하고 있으며 이를 통해서 디스커버리 구조체 정보를 송수신한다.

나. SDP 디스커버리 동작 시나리오

SDP 디스커버리 과정은 다음과 같은 세부단계로 나눌 수 있다.

a. 참여자 생성 SPDP 메시지 송수신
b. 참여자 삭제 SPDP 메시지 송수신
c. 단말개체 생성 SEDP 메시지 송수신
d. 단말개체 수정 SEDP 메시지 송수신
e. 단말개체 삭제 SEDP 메시지 송수신

모든 송수신 과정은 미리 정의된 단말개체 간에 이루어지며 a, b는 최선형(best-effort) 통신, c, d, e는 신뢰형(reliable) 통신방법을 사용한다. 본 논문에서는 세부단계 중 "c. 단말개체 생성 SEDP 메시지 송수신"의 Writer 생성에 대해서 동작 메커니즘과 관련 객체를 기술하도록 한다.

RTPS 표준명세에서는 참여자 간의 메시지 교환은 단말개체인 Writer와 Reader 간에 이루어지며 이에 대한 상세 동작을 Behavior Module로 정의하고 있다. 디스커버리 과정 또한 미리 정의된 단말개체 간에 메시지를 교환하는 과정이므로 Behavior Module의 Writer, Reader의 동작 메커니즘을 그대로 따른다. Writer와 Reader는 교환하는 정보를 CacheChange로 정의하며 이를 자신의 HistoryCache에 저장한다. 또한, 매칭된 상대편의 정보를 Writer와 Reader는 각각 ReaderProxy, WriterProxy로 만든 후 연결을 설정한다.

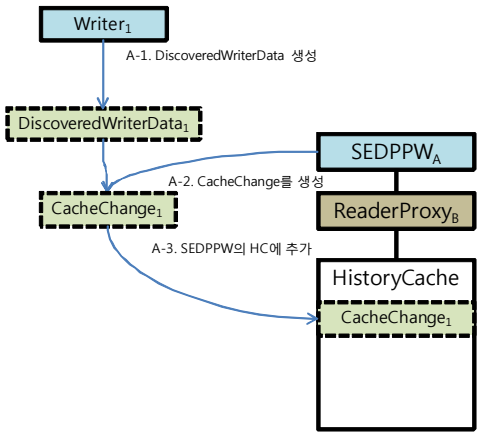

그림 3 Writer 생성측의 SEDP 동작

그림 3은 $Participant_A$에서 단말개체 중 $Writer_1$이 생성되었을 때 Writer 생성 측의 동작을 보여주고 있으며 다음의 단계를 따른다.

A-1. 응용에서는 $Writer_1$을 생성하면서 Writer의 주소, 토픽, QoS 등의 정보를 포함한 $DiscoveredWriterData_1$을 $Participant_A$에게 전달한다.

A-2. SEDPManager는 $DiscoveredWriterData_1$을 이용하여 Writer 생성에 대한 SEDP 메시지를 송신하기 위해 미리 정의된 SEDPbuiltinPublicationsWriter인 $SEDPPW_A$의 HistoryCache에 저장되는 $CacheChange_1$을

생성한다.

A-3. 생성된 CacheChange$_1$은 SEDPPW$_A$의 HistoryCache에 저장된다. SEDPPW$_A$에는 상대편 참여자 Participant$_B$의 SEDPbuiltinPublicationsReader의 정보인 ReaderProxy$_B$가 연결되어 있어 이를 이용하여 CacheChange$_1$을 전달할 수 있다.

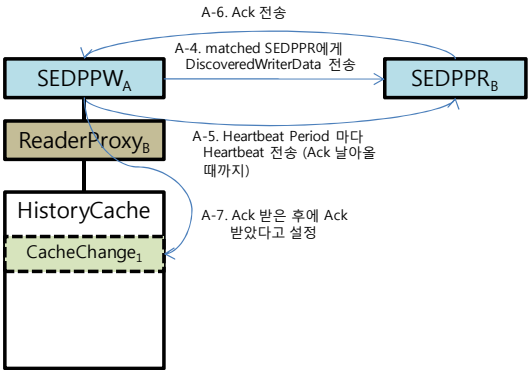

그림 4 Writer 정보 송수신시 SEDP 동작

그림 4는 그림 3에서 생성된 Writer$_1$ 정보를 송수신하는 과정에서의 동작을 보여주고 있으며 다음의 단계를 따른다.

A-4. 생성된 CacheChange$_1$을 ReaderProxy$_B$에 대응되는 Participant$_B$에게 전송한다. 전송된 정보는 Participant$_B$에서 Writer의 정보 수신을 위해 미리 정의된 단말개체인 SEDPbuiltinPublicationsReader (SEDPPR$_B$)에게 전달된다.

A-5. SEDPPW$_A$는 SEDPPR$_B$에게 정보를 전달한 후 잘 전달되었는지 확인하기 위해서 주기적으로 Heartbeat을 전송한다. Heartbeat은 SEDPPR$_B$로부터 Ack가 날아올 까지 계속 전송된다.

A-6. SEDPPR$_B$는 SEDPPW$_A$로부터 Heartbeat을 받았으면 이에 대한 응답으로 Ack를 보낸다. 만약, Writer$_1$에 대한 정보를 수신하지 못했다면 Nack를 보냄으로써 SEDPPW$_A$에게 Writer$_1$의 정보 재전송을 요청한다.

A-7. SEDPPR$_B$로부터 Ack를 받으면 History-Cache의 CacheChange$_1$에 대해서 Ack를 받았다는 설정을 해놓는다. 만약 Nack를 받았으면 A-4, A-5 과정을 반복한다.

마지막으로 그림 5는 Writer$_1$ 정보 수신 후 SEDPPR$_B$의 동작 메커니즘을 보여주고 있다.

B-1. Writer$_1$의 정보 수신 후 SEDPPR$_B$는 CacheChange$_1$을 생성한다.

B-2. 생성된 CacheChange$_1$은 자신의 History-Cache에 저장되며 SEDPPW$_A$에 대응되는 WriterProxy$_A$와 연결한다.

B-3. Participant$_B$에서 이전에 생성된 Reader 단말개체 중 Writer$_1$과 토픽 메시지 교환이 가능한 Reader를 검색한다. 토픽과 QoS 호환성 검사를 통해서 매칭이 되는 Reader$_2$가 검색되면 Writer$_1$에 대한 정보를 포함한 WriterProxy$_1$을 Reader$_2$에 연결함으로써 이후 Writer$_1$과 통신이 가능하게 한다.

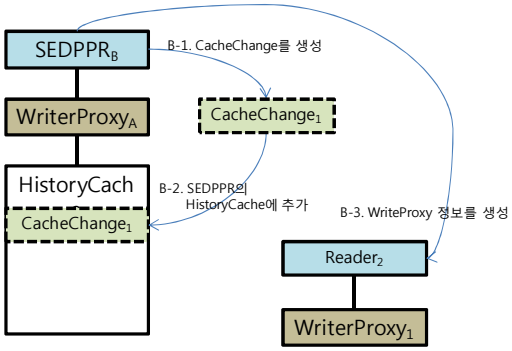

그림 5 Writer 정보 수신 후 SEDP 동작

VI. 결 론

본 논문에서는 OMG에서 데이터 중심 통신 미들웨어 표준으로 정의한 DDS의 기능 중 미들웨어 간 상호운용성 제공을 위한 핵심 기술인 디스커버리에 대해서 살펴보고 이를 구현하기 위한 설계를 하였다. RTPS 표준명세에서는 데이터 교환을 위한 메시지 포맷 및 프로토콜은 자세히 기술하고 있으나 상대적으로 디스커버리 과정은 간단히 기술하고 있다. 따라서 본 논문에서 제시한 설계 및 객체 간 동작 메커니즘은 표준 디스커버리 기능을 구현하기 위한 기초 자료로 활용될 수 있을 것이다.

참고문헌

[1] D. C. Schmidt, A. Corsaro, and H. Hag, "Addressing the Challenges of Tactical Information Management in Net-Centric Systems with DDS," The Journal of Defense Software Engineering, pp.24- 29 (2008).

[2] 권기정, 유용역, 최훈, “확장성과 효율성을 고려한 DDS 참여자 디스커버리 기법,” 한국해양정보통신학회논문지, 제13권, 제7호, pp.1344-1356 (2009).

[3] OMG, "Data Distribution Service for Real-time Systems," Version 1.2, OMG (2007).

[4] OMG, "The Real-time Publish-Subscribe Wire Protocol DDS Interoperability Wire Protocol Specification," Version 2.1, OMG (2008).