SIP기반 Presence Agent 서버 기능 설계 및 구현

현욱, 허미영, 박선옥, 강신각
*한국전자통신연구원

Design and Implementation of SIP-based Presence Agent Server

Wook Hyun, MiYoung Huh, Sunok Park, Shingak Kang
*Electronics and Telecommunications Research Institute

E-mail : whyun@etri.re.kr

요 약

Presence Agent Server는 사용자의 Presence 정보를 PUBLISH를 통해 전달 받고, SUBSCRIBE에 의해 맺어진 Subscription으로 NOTIFY 메시지를 사용하여 전달해주는 역할을 한다. IMPP 단말은 PUA(Presence User Agent)의 기능을 기본으로 가지며, 경우에 따라서는 PA(Presence Agent)의 기능까지 포함하는 경우도 있으나, Presence Agent Server는 주로 단말이 PUA의 기능만을 가지고 있을 때 유의미한 동작을 수행한다. Presence Agent Server는 오직 PA의 기능만을 담당하며, 타 Subscriber로부터의 subscription 요청을 처리한다. 그리고 확장된 개념의 Presence Agent 서버는 Resource List Event Package까지 지원을 하여, Subscriber는 단 한번의 subscription으로 N개 이상의 Presence 정보를 가지게 할 수도 있다.

본 고에서는 Presence Agent Server의 기능을 구현하기 위하여 기능별로 구현 모듈을 분리시키고 각 모듈별 기능을 기술하는 것을 목적으로 한다. 그리고 각 모듈간 연동을 위한 API를 단순화 하기 위하여 이벤트 매니저/핸들러로 분리를 시킨 방법과 정보 저장을 위한 데이터베이스 API Instruction Set을 나열 및 설명한다.

ABSTRACT

The Presence Agent Server receives user's presence information via PUBLISH request message and let subscribers to know that information. IMPP client has a functions for PUA(presence user agent) as a base operational entity. In some circumstance, the client has both pua and pa logical entities. However presence agent server has take part only of PA.

In this paper, we describes the design and implementation that contains modules based configurations and approaches for presence agent server. Also, we describe how we handle each events and database API instruction set.

키워드

SIP, IMPP, Presence Agent

Ⅰ. 서 론

Presence Agent Server는 사용자의 Presence 정보를 PUBLISH를 통해 전달 받고, SUBSCRIBE에 의해 맺어진 Subscription으로 NOTIFY 메시지를 사용하여 전달해주는 역할을 한다. IMPP 단말은 PUA(Presence User Agent)의 기능을 기본으로 가지며, 경우에 따라서는 PA(Presence Agent)의 기능까지 포함하는 경우도 있으나, Presence Agent Server는 주로 단말이 PUA의 기능만을 가지고 있을 때 유의미한 동작을 수행한다. Presence Agent Server는 오직 PA의 기능만을 담당하며, 타 Subscriber로부터의 subscription 요청을 처리한다. 그리고 확장된 개념의 Presence Agent 서버는 Resource List Event Package[1]까지 지원을 하여, Subscriber는 단 한번의 subscription으로 N개 이상의 Presence 정보를 가지게 할 수도 있다.[2]

II. Presence Agent 서버 기능 설계

이번 장에서는 XCAP 서버 기능 설계를 위하여 XCAP 서버가 가져야 할 기능과 각 기능별 블록 구성관계를 설명하도록 한다.

가. Presence Agent 서버 기능 정의

Presence Agent 서버는 여러 Presentity들로부터 presence 정보를 얻기 위하여 SIP(Session Initiation Protocol)을 사용한다. 이 서버는 presence 이벤트 패키지 및 resource list 이벤트 패키지를 이용하여 여러 Presentity들의 presence 정보를 대신하여 관리하며 PUA와의 subscription을 대행하게 된다. 이 과정에서 presentity UA에게 subscription의 발생 및 변화 정보를 전달하기 위하여 watcher-info 이벤트 패키지를 사용한다.

Presence Agent 서버를 기능별 모듈로 구성하면 아래와 같은 형태로 나누어 볼 수 있다.

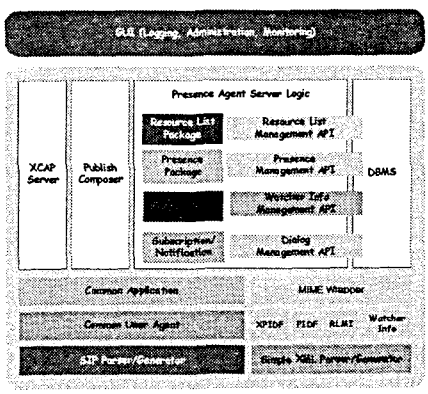

그림 1 PAS 기능별 모듈 구성

위 그림에서 제시하는 Presence Agent 서버는 IETF에서 진행중인 모든 표준문서들에 대한 기능을 제공하기 위한 블록들로써 실제 서버를 구현하고자 할 경우에는 필요에 따라 기능들을 선택하여 구현할 수 있다. 각 모듈블록별 기능 설명은 다음 장에서 다루도록 한다.

나. 모듈별 기능 정의

이 장에서는 각 모듈별로 가져야 할 기능들에 대해 설명을 한다.

- GUI

서버의 경우 별도의 GUI를 통한 조작 기능은 optional한 부분으로써 주로 Adminstration, Monitoring, Message Logging 기능등을 담당하게 되며, platform dependent 한 부분으로 볼 수 있다.

- XCAP Server

사용자의 buddy 목록 관리(black list, white list, add buddy, del buddy, …)를 담당하며, XCAP 방식으로 구동된다. XCAP은 현재 IETF에서 표준화 작업이 진행중인 상태이며, 아직 정식 RFC로 등록된 문서는 없는 상태이나, 앞으로 정식 문서로까지 진행될 것으로 보인다. Presence Agent 서버는 XCAP의 server기능만을 필요로 하며, 단말의 경우에는 XCAP Client의 기능을 구현해야 상호 연동하여 동작할 수 있다.

- Publish Composer

단말의 Publisher 기능에 대비되는 서버의 기능으로써, 단말로부터 PUBLISH를 통한 Presence Info를 전달받아 내부 Presence Database에 저장하는 기능을 담당한다. 상업적인 목적의 구현물을 제작할 경우에는 인증 기능이 필수로 들어가야 한다.

- Presence Agent Server Logic

Presence Agent 서버의 메인 로직을 담당하는 부분으로써 Subscription 및 Notification 에 대한 권한 검증 및 Presence 데이터 전달등을 담당한다.

- Resource List Package

Resource List Event Package 기능을 지원하기 위한 로직을 담고 있으며, 메시지 생성을 비롯한 수신한 경우의 프로세싱, 이벤트가 발생할 경우의 동작을 담당한다.

- Presence Package

가장 기본적으로 가져야 할 기능으로써 presence 이벤트에 대한 처리를 담당한다.

- WatcherInfo Package

자신에 대해 누가 Subscribe를 하고자 하는 등의 기능을 제공해 주는 패키지이다.

- Subscription/Notification Engine

RFC3265에 정의된 Subscription/Notification의 기능을 담당하는 모듈이다.

- Common Application

RFC3261 기반 SIP 트랜잭션을 관리하는 CUA의 기능을 활용하기 위한 Application Interface로써 범용의 API를 제공한다.

- Common User Agent

RFC3261 기반 SIP 트랜잭션을 관리하며, 트랜잭션 단위의 범용 API를 제공한다.

- SIP Parser/Generator

RFC3261 SIP 메시지를 파싱하고 생성해내는 기능을 가지는 SIP Stack.

- Presence Database

Presence정보를 비롯하여, Subscription, DialogInfo등에 대한 정보를 담고 있는 데이터베이스 모듈이며, 범용으로 사용하기 위하여 DB specific한 함수들을 별도로 모아 놓고, 범용 API를 제공한다.

- XPIDF

Presence 정보를 전달하는 방식중 하나인 XPIDF 포맷을 지원하기 위한 파싱 및 생성 엔진이다. XPIDF는 정식으로 표준화되지는 않고 있으나, MSN에서 사용되는 포맷으로써 상호 연동 기능 점검을 위해 만들어진 temporary 모듈이다.

- PIDF

IETF에 표준화중인 Presence 정보 표현 방식으로써, CPIM의 형태로 제공된다. 이 모듈은 CPIM 문서의 파싱 및 생성을 담당한다.

- RLMI

XML로 Resource List를 표현하기 위한 방식으로써 현재 IETF에서표준화 진행중에 있으며, 이 모듈에서는 파싱 및 생성 기능을 담당한다.

III. XCAP 서버 구현

아래 그림은 전체 Presence Server의 구조를 도시한 것으로써 크게 Presence Event Manager와 Event Handler, DB Interface로 나뉘어 지도록 구성되어 있음을 표현한다.

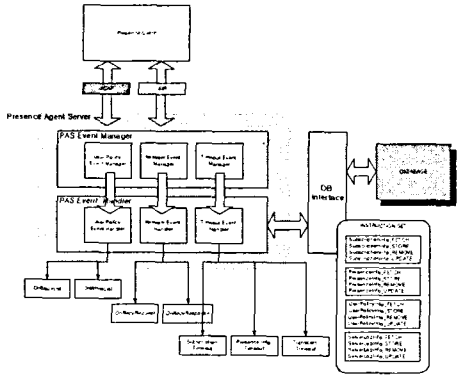

그림 2 PAS logic

가. Pas Event Manager

PasEventManager는 사용자 정책 변경 이벤트를 관리하는 UserPolicyEventManager와 네트워을 통해 수신된 신호를 전달해주는 NetworkEventManager, 데이터베이스내의 사용자 상태정보 및 트랜잭션의 타임아웃이 발생한 경우 해당 이벤트를 탐지하는 세 개의 이벤트 관리자들로 구성되며 각 이벤트에 따라 적당한 처리자(Handler)를 호출하며, 각 EventManager들은 해당되는 이벤트 핸들러에게 처리를 넘기게 되며, 이 장의 나머지 부분에서 각 핸들러의 동작을 설명하도록 한다.

나. UserPolicyEventHandler

아래 그램은 UserPolicyEvent가 발생되는 상황을 도시한 것이다.

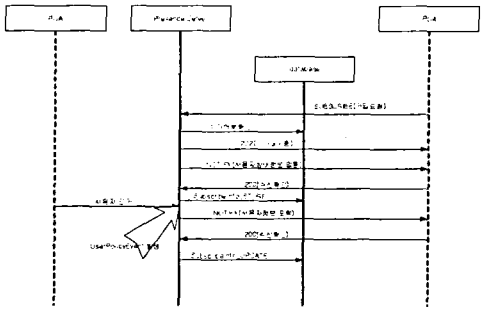

그림 3 UserPolicyEvent 처리 과정

UserPolicyEvent가 발생되었을 경우 프리젠스 서버는 이벤트 파라미터로 넘겨진 사용자 결정이 허락인지 불허인지에 따라 다른 동작을 수행한다.

사용자의 결정이 허락일 경우 UserPolicyEventHandler내의 OnWhiteList 모듈이 호출되며, 그 동작은 다음과 같다.

① 이벤트의 발생을 탐지한다.
② 데이터 베이스에 UserPolicy_STORE 명령을 호출하여 해당 사용자에 대한 인가 정보를 기록한다.
③ 데이터 베이스에서 SubscriptionInfo_FETCH 명령을 호출하여 현재 관계된 트랜잭션을 가져온다.
④ 관계된 트랜잭션에 대해 현재 사용자 상태 정보가 포함된 NOTIFY 메시지를 전송한다.
⑤ NOTIFY 메시지를 잘 수신했다는 의미의 200 응답을 수신할 때 까지 재전송한다.
⑥ 200 응답을 수신하면 재전송을 중단하고, SubscriptionInfo_UPDATE 명령을 호출하여 최신 트랜잭션 정보로 데이터베이스내의 값을 갱신한다.

다. NetworkEventHandler

네트웍상의 이벤트인 SIP 메시지가 수신된 경우의 이벤트를 전달받아 아래와 같은 절차에 따라 동작한다.

① 네트웍 이벤트 발생을 탐지한다.
② 수신한 SIP 메시지의 종류가 Request인지 Response인지 판단한다.
③ 메시지의 종류에 따른 처리를 수행한다.

라. TimeoutEventHandler

타임아웃 이벤트는 크게 트랜잭션 유효시간이 경과한 경우와 사용자 정보의 유효시간이 경과한 경우, 메시지를 전송하고 난 후 그에 따른 응답을 일정 시간이내에 수신하지 못하는 경우 발생하는

세 가지 경우로 나누어 볼 수 있다.

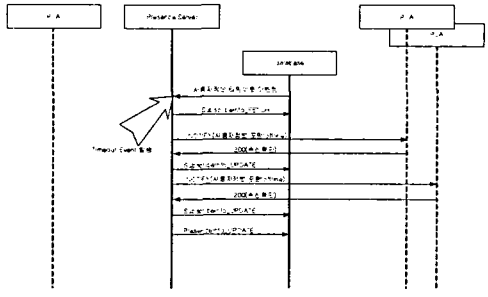

그림 4 PresenceInfo Timeout Event 처리 과정

위 그림은 사용자 정보가 timeout이 되기전까지 refresh 메시지를 받지 못하게 될 경우 발생되는 상황을 도식화한 것으로써 subscriber에게 해당 정보가 더 이상 유효하지 않음을 의미하는 NOTIFY를 전달한다.

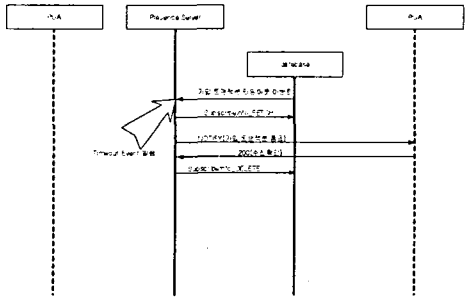

그림 5 Subscription Timeout 처리과정

위 그림은 추가적인 Subscription이 timeout이 발생한 경우의 동작을 도식화한 것으로써 해당 Subscriber에게 NOTIFY를 전달함으로써 subscription의 종료를 통보한다.

아래 그림은 Presentity의 presence 정보가 변경되었을 경우 Subscriber에게 해당 정보를 전달하기 위한 NOTIFY를 송신하였으나 상대로부터 response가 도달하지 않음으로 인한 transaction timeout이 발생한 경우를 도식화한 것이다.

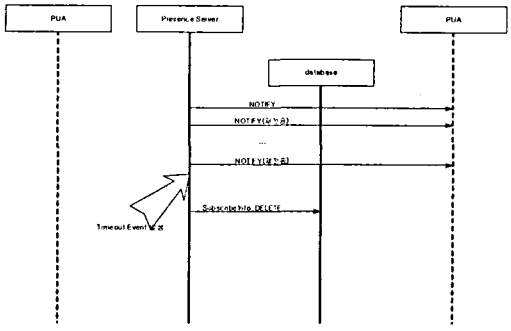

그림 6 Transaction Timeout Event 처리과정

PUA의 transaction layer에서는 상대로부터 response를 수신할 때까지 최대 10회의 재전송을 시도한다.

표 1 Database API Instruction SET

구분	Primitive
Subscription 관리	SubscriptionInfo_FETCH
	SubscriptionInfo_STORE
	SubscriptionInfo_REMOVE
	SubscriptionInfo_UPDATE
PresenceInfo 관리	PresenceInfo_FETCH
	PresenceInfo_STORE
	PresenceInfo_REMOVE
	PresenceInfo_UPDATE
UserPolicy 관리	UserPolicyInfo_FETCH
	UserPolicyInfo_STORE
	UserPolicyInfo_REMOVE
	UserPolicyInfo_UPDATE

V. 결 론

본 고는 SIP기반 프레즌스 서비스를 제공하기 위한 프레즌스 서버의 기능을 제공하기 위한 설계 및 구현 사항에 대하여 논하였다. 또한 서버의 안정성 및 대용량 subscription을 지원하기 위하여 데이터베이스 기반으로 동작한다.

참고문헌

[1] IETF, draft-ietf-simple-event-list-07.txt, "A Session Initiation Protocol (SIP) Event Notification Extension for Resource Lists"

[2] IETF, RFC3856, "A Presence Event Package for the Session Initiation Protocol"