프레즌스 메시지 트래픽 절감 표준 기술 동향

Analysis on Presence Message Traffic Reduction Standardization

허미영 (M.Y. Huh) 통합망표준연구팀 선임연구원
박선옥 (S.O. Park) 통합망표준연구팀 연구원
현 욱 (W. Hyun) 통합망표준연구팀 연구원
강신각 (S.G. Kang) 통합망표준연구팀 팀장

목 차

Ⅰ. 서론
Ⅱ. 프레즌스 모델
Ⅲ. SIP 확장 프로토콜 표준들
Ⅳ. 프레즌스 메시지 트래픽을 줄이기 위한 표준 기술들
Ⅴ. 프레즌스 정보 처리에 대한 전체 동작 흐름
Ⅵ. 결론

메신저로 대표되는 IMPP 서비스의 핵심 기술로 프레즌스 프로토콜이 필요하며, SIP 기반 프레즌스 프로토콜에 대한 표준화 작업이 IETF SIMPLE WG에서 진행중에 있다. 이러한 프레즌스 서비스는 앞으로 인터넷상에서 핵심 통신 기술로 사용되어 프레즌스 기반 통합된 서비스가 많이 등장하리라 예상된다. 그러나, SIP 기반 프레즌스 서비스에서 간단한 상태 정보 등의 변화를 전달하기 위하여 전달되는 프레즌스 메시지 트래픽의 양이 상당히 많은 것이 문제점으로 대두되었다. 이는 대역폭이 제한된 무선 환경의 경우 특히 많은 문제를 야기한다. 따라서, 본 고에서는 SIP 기반 프레즌스 프로토콜 관련 표준화 현황을 통한 기본 개념을 살펴보고 프레즌스 메시지 트래픽을 줄이기 위한 다양한 방법들에 대한 최근 표준화 동향을 살펴보고자 한다.

Ⅰ. 서론

메신저로 대표되는 IMPP 서비스는 개인용이나 업무용 통신 수단으로 우리 생활에 깊숙히 자리를 잡아가고 있다. 이러한 IMPP 서비스를 제공하기 위한 핵심에는 서로 다른 IMPP 서비스 가입자나 제공자간에 프레즌스 정보에 대하여 주고받기 위한 표준화된 프레즌스 프로토콜이 필요하다.

IETF SIMPLE WG에서는 프레즌스 프로토콜을 SIP을 확장하여 정의하고자 구성된 그룹이다[1], [2]. SIMPLE WG은 SIP가 3GPP나 NGN 등의 환경에서 시그널링 프로토콜로 채택되고, SIP가 확장성면에서 장점이 있어 통합된 다양한 서비스를 제공할 수 있다는 기대감으로 관심이 집중되고 있다. 앞으로는 프레즌스 서비스가 인터넷상에서 통신의 핵심 기술로 사용되어 프레즌스 기반 통합된 서비스가 많이 등장하리라 예상된다. 그러나, SIP 기반 프레즌스 서비스에서 간단한 상태 정보 등의 변화를 표현하기 위하여 전달되는 프레즌스 메시지 트래픽의 양이 상당히 많은 것이 문제점으로 대두되었다. 이는 대역폭이 제한된 무선 환경의 경우 특히 많은 문제를 야기한다. 또한, 무선 환경뿐만 아니라 인터넷 환경에서도 변경된 프레즌스 정보를 가입자에게 무조건 전달하는 것보다 원하는 사용자에게 원하는 정보만을 전달함으로써 맞춤형 서비스를 제공하는 것이 사용자에게 좀더 유용한 서비스가 될 수 있다.

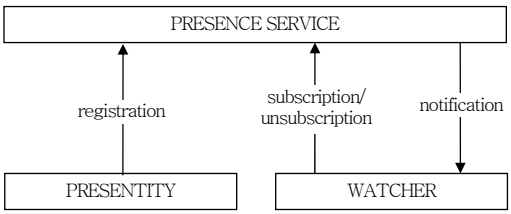
본 고에서는 IMPP를 위한 SIP 확장 프로토콜 관련해, Ⅱ장에서 IETF IMPP WG에서 정의했던 프레즌스 모델에 대하여 살펴보고[3], Ⅲ장에서 프레즌스 모델에 따라 현재 RFC로 정의된 SIP 확장 프로토콜 표준들에 대하여 대략적으로 살펴보고, Ⅳ장에서 전달되는 프레즌스 정보를 줄이기 위하여 IETF SIMPLE WG 표준에서 개발하고 있는 방법들에 대하여 살펴보고, Ⅴ장에서 프레즌스 정보의 생성에서부터 가입자에게 최종 통지하기까지의 프레즌스 정보 처리를 위한 전체적인 처리 과정에 대하여 간단히 살펴보도록 하겠다.

Ⅱ. 프레즌스 모델

SIMPLE WG에서 정의하는 SIP 확장 표준들은 RFC2778에서 정의한 프레즌스 모델에 근거하고 있다[4]. 프레즌스 모델을 간단하게 살펴보면 (그림 1)과 같다.

프레즌스 모델은 크게 프리젠티티(presentity), 프레즌스 서비스(presence service), 와처(watcher)로 구성된다. 프리젠티티는 프레즌스 정보를 제공하는 소스 역할을 한다. 프레즌스 서비스는 프리젠티티에서 제공하는 프레즌스 정보를 수용하여 저장하고 있으며 프레즌스 정보를 요구하는 와처에게 프레즌스 정보를 전달하는 역할을 한다. 와처는 프레즌스 서비스에게 프레즌스 정보를 요구하여 수신하는 역할을 한다.

프레즌스 모델에서 정의하는 프레즌스 정보는 다양한 장치를 통하여 통신하기 위한 사용자의 능력 및 의지라고 정의하고 있다. 이는 하나 이상의 프레즌스 튜플(tuple)로 구성되며, 프레즌스 튜플은 OPEN, CLOSE와 같은 상태나 인스턴트 메시징 주소와 같은 통신 주소 등을 포함한다.



(그림 1) 프레즌스 모델

Ⅲ. SIP 확장 프로토콜 표준들

프레즌스 모델에 따라 개발된 SIP 확장 프로토콜 표준은 기본적으로 RFC3261 SIP 표준의 기본 동작을 위배하지 않도록 하고 있다. 현재 프레즌스 서비스를 위하여 기본 표준으로 개발된 것들은 다음과 같다.

1. 프레즌스 정보관련 표준화 동향

가. 프레즌스 정보 데이터 포맷

IMPP 모델에서 정의하는 프레즌스 정보를 표현하기 위한 공통 데이터 포맷으로 PIDF가 정의되었으며, 이는 IETF IMPP WG에서 RFC3863 표준으로 제정하였다[5]. PIDF는 MIME 타입으로 "application/pidf+ xml"으로 콘텐트 타입이 정의되어 있다. (그림 2)는 RFC3863에서 정의된 PIDF XML 스키마이다.

PIDF에 대한 XML 스키마에서 보면 'entity' 부분에 해당 프리젠티티에 대한 URI를 지정하고, 'note' 부분에 프레즌스 정보에 대하여 사람이 이해할 수 있는 내용이 포함된다. 각 'tuple' 정보는 SIP에서 정의하는 각 인스턴스를 표현하며 각 인스턴스를 구분하기 위하여 'id' 부분이 사용되고, 'status' 부분에 주어진 서비스에 대한 유용성에 대한 내용을 표현하고, 'contact' 부분에 통신하기 위해 사용될 사용자를 위한 접촉 정보를 표현하고 있다. 'time-stamp'에는 프레즌스 정보가 생성된 시간 정보를 나타낸다.

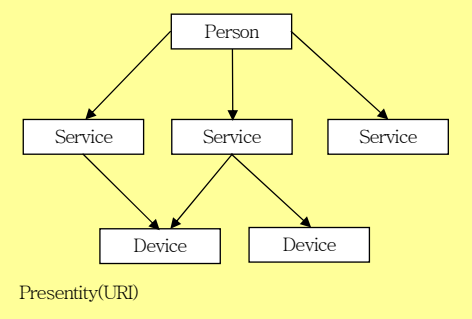
```
<?xml version="1.0" encoding="UTF-8"?>
  <xs:schema targetNamespace="urn:ietf:params:xml:ns:pidf"
      xmlns:tns="urn:ietf:params:xml:ns:pidf">
      ....
    <xs:element name="presence" type="tns:presence"/>

    <xs:complexType name="presence">
      <xs:sequence>
        <xs:element name="tuple" type="tns:tuple"
                minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="note" type="tns:note"
                minOccurs="0" maxOccurs="unbounded"/>
        <xs:any namespace="##other" processContents="lax"
                minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="entity" type="xs:anyURI" use="required"/>
    </xs:complexType>

    <xs:complexType name="tuple">
      <xs:sequence>
        <xs:element name="status" type="tns:status"/>
        <xs:any namespace="##other" processContents="lax"
                minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="contact" type="tns:contact" minOccurs="0"/>
        <xs:element name="note" type="tns:note" minOccurs="0"
                maxOccurs="unbounded"/>
        <xs:element name="timestamp" type="xs:dateTime"
                minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="id" type="xs:ID" use="required"/>
    </xs:complexType>
    ....
```

(그림 2) PIDF XML 스키마

나. 데이터 모델

RFC3863에서 정의한 PIDF 포맷이 IMPP WG에서 정의한 프레즌스 정보에 대한 개념을 충분히 표현하는 데 부족하다는 인식이 있다. 또한, PIDF에서 다루고 있는 정보가 여러 응용 서비스 분야에서 활용되기 위해서는 다양한 형태로 확장이 필요한다. 따라서, 향후 확장될 프레즌스 정보 데이터가 프레즌스 정보에 대한 개념을 충분히 잘 표현하고, 다양한 통신 시스템과 일관성있게 매핑될 수 있도록 유도하기 위하여 데이터 모델을 표준화하고 있다.

(그림 3)은 데이터 모델에서 정의하는 기본 컴포넌트와 프리젠티티 내에서 그들 사이의 관계를 보여준다. Data model은 person, service, device 데이터 컴포넌트가 있고, 한 프리젠티티 URI에 대하여 이들 데이터 컴포넌트 사이의 관계에 의하여 세부정보가 결합되어 표현된다. 이는 IETF SIMPLE WG에 "Data Model for Presence"란 제목으로 제안되어 현재 드래프트 상태이다[6].



(그림 3) 데이터 모델

다. 확장된 프레즌스 데이터 포맷

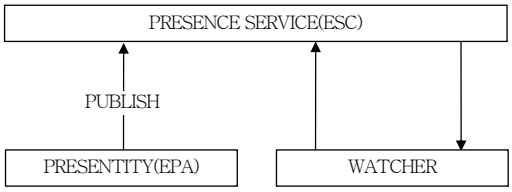
확장된 데이터 포맷의 한 예로 activities, class, mood, place-type, relationship, status-icon, user-input 등을 추가한 RPID 표준이 있다. 이는

PIDF의 노트 속성만으로는 캘린더, 센서, 위치 추적 등을 통한 자동 입력 정보 도출이 부족하여 상업적인 용도로 활용하는 데 제한이 발생하기 때문에 제안된 것이다. 앞에서 표준화 작업을 하고 있는 data model에 근거하여 추가적인 프레즌스 속성 정보를 정의하고 있다. 기존 "application/pidf+ xml"와 backward compatibility를 보장하기 위해 동일한 콘텐트 타입을 사용한다. 이는 IETF SIMPLE WG에 "RPID: Rich Presence: Extensions to the PIDF"로 제안되어 현재 드래프트 상태이다[7].

2. 프레즌스 정보 등록 관련 표준화 동향

프레즌스 정보에 대한 소스로서 프리젠티티에서는 프레즌스 정보를 프레즌스 서비스에 등록하는 과정이 필요하다. IETF SIP WG에서는 프레즌스 정보 등록을 위한 SIP 확장 표준으로 RFC3903을 제정하였다[8]. 이 표준에서는 적절한 이벤트 패키지를 위한 이벤트 상태를 전달하는 것을 목적으로 PUBLISH 메소드를 정의하였다. 또한, 이 표준에서는 EPA와 ESC를 정의하고 있다. EPA는 PUBLISH 메시지를 발생시키는 UAC이고, ESC는 PUBLISH 메시지를 처리하는 UAS이다. 프레즌스 모델에서는 (그림 4)와 같이 프리젠티티가 EPA에 해당되고, 프레즌스 서비스가 ESC에 해당된다.

PUBLISH 메소드는 RFC3261에서 정의하는 REGISTER와 유사하다. 동일한 메소드를 이용하여 초기 생성, 내용 수정, 기간 변경, 삭제 등이 가능하다. 단, 초기 생성을 제외하고 ESC에서 응답 메시지의 SIP-ETag에 제공한 entity-tag를 SIP-If-Match 헤더에 포함시키는 방식을 사용함으로써 ESC에서 허용된 사용자만이 프레즌스 정보를 등록할 수 있도록 한다. PUBLISH 메소드 내에 포함되는 바디는 이벤트 패키지에 따라 좌우되며, 프레즌스 정보를 위해서는 RFC3863에서 정의한 "application/pidf+ xml"콘텐트 타입이 사용된다.



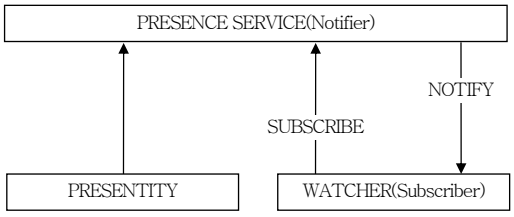
(그림 4) 프리젠티티에서 프레즌스 서비스로의 프레즌스 정보 등록

3. 프레즌스 정보 가입 및 통지 관련 표준화 동향

가. 가입 및 통지 관련 프레임워크

프레즌스 정보 가입 및 통지를 위한 SIP 확장 표준은 IETF SIP WG에서 RFC3265로 제정하였다[9]. 이 표준에서는 적절한 이벤트에 대한 가입 및 이벤트 상태에 대한 통지를 위한 프레임워크를 정의한다. 이를 위하여 SUBSCRIBE와 NOTIFY 메소드를 정의하였다. 또한, 이 표준에서는 가입자(subscriber)와 통지자(notifier)를 정의하고 있다. 가입자는 특정 프리젠티티에 대한 가입을 요청하기 위한 SUBSCRIBE 메시지를 발생시키고, 통지자로부터 가입자가 가입한 프리젠티티의 상태에 관한 정보를 포함하고 있는 NOTIFY 메시지를 수신하는 UA이다. 통지자는 SUBSCRIBE 메시지를 수신하여 가입자가 요구한 프리젠티티의 상태에 대한 통지 메시지인 NOTIFY를 생성하는 UA이다. 프레즌스 모델에서는 (그림 5)와 같이 와처가 가입자에 해당되고, 프레즌스 서비스가 통지자에 해당된다. RFC 3265에서는 가입자와 통지자의 동작에 대하여 자세히 기술하고 있다.

또한, RFC3265에서는 프레임워크 표준으로 이를 바탕으로 각 이벤트 패키지를 정의할 수 있도록



(그림 5) 와처와 프레즌스 서비스 사이의 가입 및 통지

하고 있다. 이벤트 패키지는 특정 클래스의 이벤트에 대하여 프레임워크를 기반으로 구체적인 응용을 정의하는 것으로, 통지자에 의하여 가입자에게 알려줄 상태 정보의 집합 등을 정의한다. 이러한 상태 정보를 전달하기 위하여 프레임워크를 기반으로 한 문법과 의미를 각 이벤트 패키지에서는 정의할 수 있도록 하고 있다.

나. 프레즌스 패키지

현재 IETF SIMPLE WG에서 정의한 이벤트 패키지로 RFC3856에서 정의한 프레즌스 패키지(presence event package)가 있다[10]. 가입자는 SUBSCRIBE 메시지의 Event 헤더에 "presence"를 지정함으로써 프리젠티티의 프레즌스 정보를 요구한다. 원하는 프리젠티티에 대한 정보는 SUBSCRIBE 메시지의 Request-URI에 표현된다. 통지자를 통해 가입자에게 전달되는 프레즌스 정보는 RFC3863에서 정의하는 PIDF 포맷을 따르고 있다.

이 프레즌스 패키지는 관심있는 프리젠티티의 프레즌스 정보를 요청하고 통지받는 가장 기본이 되는 패키지이다.

다. 와처 정보 패키지

프레즌스 정보의 소스에 해당되는 프리젠티티는 자신의 프레즌스 정보를 프레즌스 서비스에게 제공한다. 프레즌스 서비스는 임의의 와처로부터 특정 프리젠티티의 프레즌스 정보 요청에 대하여 정해진 인증과 인가 과정을 거친 후 요청한 프리젠티티에 대한 프레즌스 정보를 제공하게 된다. 이 과정에서 프레즌스 정보의 소스인 프리젠티티는 누구에게 자신의 프레즌스 정보를 제공할 것인지를 지정할 수 있어야 한다.

이를 위하여 프리젠티티는 누가 자신의 프레즌스 정보를 요청하는지에 대한 정보를 알고 있어야 한다. 즉, 프리젠티티 자신에 대한 와처 정보를 알고 있어야 한다. 이러한 목적으로 정의된 것이 와처 정보 패키지이다. 현재 IETF SIMPLE WG에서 RFC 3857로 와처 정보 패키지(watcher information event template-package)를 정의하고 있으며, 와처 정보를 전달하기 위한 포맷으로 RFC3858을 정의하고 있다[11],[12].

이 와처 정보 패키지는 임의의 다른 패키지와 연관되어 표현되는 템플릿 패키지로 SUBSCRIBE 메시지의 Event 헤더에 연관된 상위 패키지 명에 ".winfo"를 추가함으로써 프레즌스 패키지와 구분할 수 있도록 한다. 프레즌스 패키지에 대한 와처 정보 템플릿 패키지는 "presence.winfo"로 표현된다.

패키지 내에서 정의하는 와처 정보는 특정 이벤트 패키지 내 특정 리소스에 가입된 사용자의 집합으로 사용자가 가입시나 가입 탈퇴시, 가입에 대한 허용시나 금지시 등에 대한 정보를 포함하고 있으며, 와처 정보 가입자는 이러한 정보에 대하여 가입하여 이들 정보에 대한 변화 발생시 통지를 받을 수 있게 된다.

(그림 6)은 와처에 대한 정보를 담고 있는 NOTIFY 메시지의 예이다. Sip:joe@example.com이란 와처 정보 가입자가 자신에 대한 와처 정보를 알기 위하여 "presence.winfo"로 와처 정보 가입을 한 상태이다. (그림 6)은 첫번째 전달된 통지 메시지로 sip:A@example.com이란 와처가 SUBSCRIBE 메시지를 통해 가입 요청을 하였으나 가입 허용을 하지 않아 pending 상태임을 알려준다. 이러한 통지

```
NOTIFY sip:joe@pc34.example.com SIP/2.0
Via: SIP/2.0/UDP server19.example.com;branch=z9hG4bKnasaii
From: sip:joe@example.com;tag=xyzygg
To: sip:joe@example.com;tag=123aa9
Call-ID: 9987@pc34.example.com
CSeq: 1288 NOTIFY
Contact: sip:server19.example.com
Event: presence.winfo
Max-Forwards: 70
Content-Type: application/watcherinfo+xml
Content-Length: ...

<?xml version="1.0"?>
<watcherinfo xmlns="urn:ietf:params:xml:ns:watcherinfo"
      version="0" state="full">
<watcher-list resource="sip:joe@example.com" package="presence">
<watcher id="77ajsyy76" event="subscribe" status="pending">
      sip:A@example.com</watcher>
</watcher-list>
</watcherinfo>
```

(그림 6) 와처 정보 통지의 예

메시지를 받은 와처 정보 가입자는 이후 sip:A@example.com에게 가입 허용 여부를 지정하는 동작을 취할 수 있다.

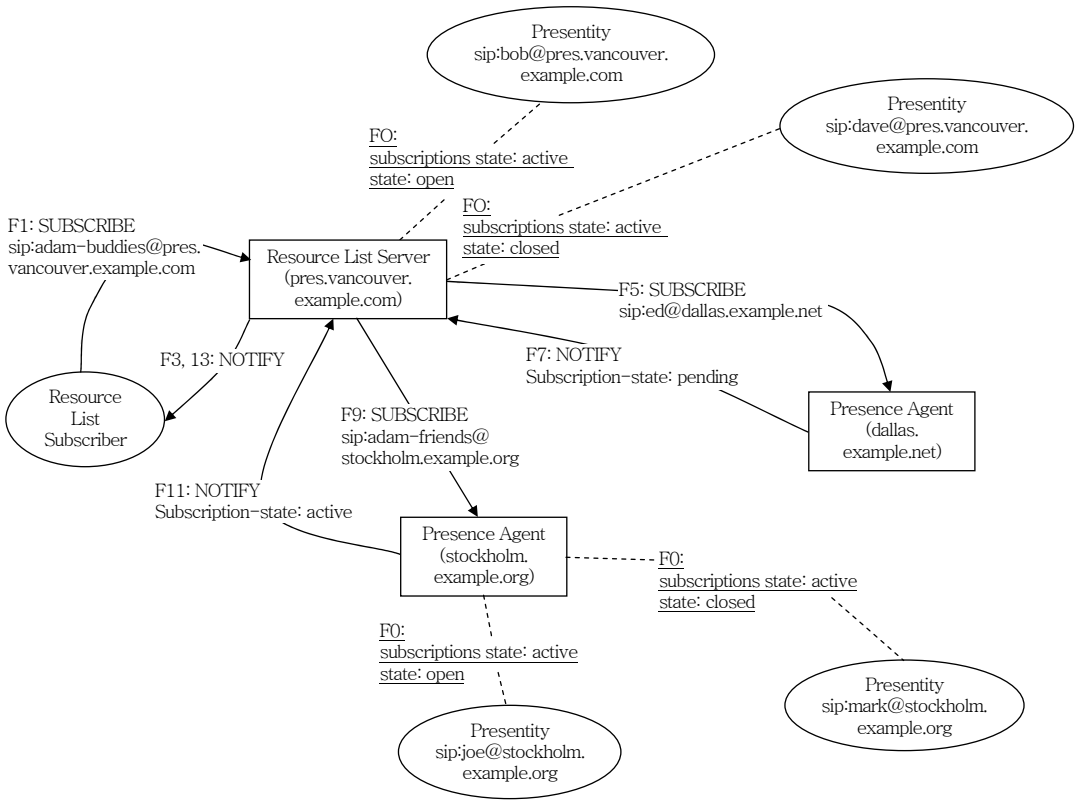
한편, 프리젠티티는 프레즌스 서비스에게 특정 와처가 자신에게 가입을 요구시 이의 허용 여부 및 허용 정도를 지정할 수 있어야 한다. 이에 대한 지정 방법으로 DIAMETER, XCAP pres-rules usage 등이 가능하다[13],[14].

Ⅳ. 프레즌스 메시지 트래픽을 줄이기 위한 표준 기술들

SIMPLE 관련 기술은 SIP를 기반으로 하고 있으며, XML을 바디 정보로 표현함으로써 다양한 응용이나 서비스에 맞게 확장할 수 있도록 하는 장점이 있다. 그러나, 간단한 상태 정보 등의 변화를 표현하는 데에도 전달되는 트래픽의 양이 상당히 많다. 이는 대역폭이 제한된 무선 환경의 경우 많은 문제를 야기한다. 따라서, 본 장에서는 프리젠티티, 프레즌스 서비스, 와처 사이에 전달되는 프레즌스 정보를 줄이기 위하여 IETF에 제안되어 표준화되고 있는 방법들에 대하여 소개한다.

1. 리소스 리스트 확장 기술

일반적으로 가입자는 관심있는 여러 프리젠티티에 동시에 가입을 하게 된다. 이는 RFC3856 프레즌스 패키지에서 정의하는 방식에 따라 프리젠티티에 각각 가입을 하는 것이 된다. 리소스 리스트 확장 기술은 가입자가 각 프리젠티티에게 개별적으로 가입을 하는 것보다 관심있는 프리젠티티의 목록을 트리 구조의 리스트 형태로 만들어 두고, 이 리스트에 한 번에 가입함으로써 리소스 리스트 내의 리소스 중



(그림 7) 리소스 리스트 가입에 대한 RLS의 처리 동작 흐름

어느 하나라도 상태 변화가 있을 때나 여러 리소스의 상태 변화를 모아서 함께 통지를 받음으로써 빈번한 트래픽의 양을 줄이고자 하는 방법에 대한 것이다. 현재 IETF SIMPLE WG에 "SIP Event Notification Extension for Resource Lists"로 제안되어 현재 드래프트 상태에 있다[15].

```
F3. Local RLS -> Terminal
NOTIFY sip:terminal.vancouver.example.com SIP/2.0
Via: SIP/2.0/TCP pres.vancouver.example.com;branch=z9hG4bKMgRenTETmm
Max-Forwards: 70
From: <sip:adam-buddies@pres.vancouver.example.com>;tag=zpNctbZq
To: <sip:adam@vancouver.example.com>;tag=ie4hbb8t
Call-ID: cdB34qLToC@terminal.vancouver.example.com
CSeq: 997935768 NOTIFY
Contact: <sip:pres.vancouver.example.com>
Event: presence
Subscription-State: active;expires=7200
Require: eventlist
Content-Type: multipart/related;type="application/rlmi+ xml";
        start="<nXYxAE@pres.vancouver.example.com>"; boundary="50UBfW7LSCVLtggUPe5z"
Content-Length: 1560

--50UBfW7LSCVLtggUPe5z
Content-Transfer-Encoding: binary
Content-ID: <nXYxAE@pres.vancouver.example.com>
Content-Type: application/rlmi+ xml;charset="UTF-8"

<?xml version="1.0" encoding="UTF-8"?>
<list xmlns="urn:ietf:params:xml:ns:rlmi"
      uri="sip:adam-friends@pres.vancouver.example.com" version="1" fullState="true">
  <name language="en">Buddy List at COM</name>
  <name language="de">Liste der Freunde an COM</name>

<resource uri="sip:bob@vancouver.example.com">
    <name>Bob Smith</name>
    <instance id="juwigmtboe" state="active"
              cid="bUZBsM@pres.vancouver.example.com"/> </resource>
<resource uri="sip:dave@vancouver.example.com">
    <name>Dave Jones</name>
    <instance id="hqzsuxtfyq" state="active"
              cid="ZvSvkz@pres.vancouver.example.com"/> </resource>
<resource uri="sip:ed@dallas.example.net">
    <name>Ed at NET</name> </resource>
<resource uri="sip:adam-friends@stockholm.example.org">
    <name language="en">My Friends at ORG</name>
    <name language="de">Meine Freunde an ORG</name> </resource> </list>

--50UBfW7LSCVLtggUPe5z
Content-Transfer-Encoding: binary
Content-ID: <bUZBsM@pres.vancouver.example.com>
Content-Type: application/pidf+ xml;charset="UTF-8"

<?xml version="1.0" encoding="UTF-8"?>
<presence xmlns="urn:ietf:params:xml:ns:pidf"
    entity="sip:bob@vancouver.example.com">
  <tuple id="sg89ae">
    <status> <basic>open</basic> </status>
    <contact priority="1.0">sip:bob@vancouver.example.com</contact>
  </tuple> </presence>
--50UBfW7LSCVLtggUPe5z
Content-Transfer-Encoding: binary
Content-ID: <ZvSvkz@pres.vancouver.example.com>
Content-Type: application/pidf+ xml;charset="UTF-8"

<?xml version="1.0" encoding="UTF-8"?>
<presence xmlns="urn:ietf:params:xml:ns:pidf"
    entity="sip:dave@vancouver.example.com">
  <tuple id="slie74">
    <status> <basic>closed</basic> </status> </tuple> </presence>

--50UBfW7LSCVLtggUPe5z—
```

(그림 8) 리소스 리스트 확장에 따른 통지의 예

가입자는 리소스 리스트를 생성하고, 리소스 리스트 URI에 SUBSCRIBE를 전송함으로써 관심있는 프리젠티티 목록에 가입을 한다. 리소스 리스트는 URI로 0개 이상의 리소스를 포함하는 리스트이며 트리 구조를 취하고 있다. 리소스는 가입할 상태를 갖고 있는 논리적인 엔티티로 프리젠티티거나 다른 리소스 리스트일 수 있다.

리소스 리스트에 대한 가입 요청을 받아 처리하는 통지자는 RLS라 불리며, 가입자에 대한 인증과 인가 과정을 거쳐 리소스 리스트 내 각 리소스에 대한 프레즌스 정보를 모아 가입자에게 전달하는 역할을 한다. 이때, 리소스 리스트 내 리소스의 상태 변화에 대하여 통지를 버퍼링하여 트래픽의 양을 줄일 수도 있다.

리소스 리스트 서버가 통지시 트리 구조의 리소스 리스트에 대한 정보를 표현하기 위하여 "application/rlmi+ xml" 콘텐트 타입을 지원하도록 한다. 이 포맷은 첫번째 부분에서 리스트에 관한 메타 정보로 멤버십이나 하위

리소스에 대한 가입 상태 등을 포함하고 있고, 나머지 부분에서 메타 정보 내에 나열된 리소스의 실제 상태를 전달한다.

RLS가 통지시 NOTIFY의 바디 정보에 표현되는 정보는 처음에는 모든 정보(full state)를 포함하나 이후에는 이전에 보낸 것에서 변경된 부분 정보(partial state)만을 전달한다. 이에 대해 리소스 리스트 가입자는 각 리소스 리스트에 대한 테이블을 유지하여 관련 정보의 연관성을 유지하여 처리한다.

(그림 7)은 가입자가 sip:adam-friends@pres.vancouver.example.com이란 리소스 리스트 URI에 가입시 pres.vancouver.example.com의 RLS가 처리하는 동작을 표현한 것이다. RLS는 해당 리소스 리스트 URI 내 리소스로 자신이 관리하는 도메인의 sip:bob@vancouver.example.com, sip:dave@vancouver.example.com과 자신이 관리하지 않는 sip:ed@dallas.example.net, sip:adam-friends@stockholm.example.org를 포함하고 있음을 인지하고, 자신이 관리하고 있지 않는 리소스에 대한 프레즌스 정보 요구를 하는 것과 함께 가입자에게 자신이 알고 있는 정보를 모아 즉시 NOTIFY를 보내고 이후 다른 리소스 정보를 알게 되면 다시 가입자에게 NOTIFY를 전송하는 과정을 보여준다. (그림 8)은 (그림 7)의 F3 NOTIFY 메시지에 대하여 보여주고 있다. F3 메시지 내에는 RLS에서는 자신이 관리하는 bob과 dave에 대한 프레즌스 정보로 open과 closed 상태 정보도 포함하고 있다는 것을 알 수 있다.

한편, 리소스 리스트 가입자는 프레즌스 서비스에게 리소스 리스트 URI의 세부 구성이 어떻게 되는지에 대하여 지정할 수 있어야 한다. 이에 대한 지정 방법으로 XCAP rls-services usage 등이 가능하다[16].

2. 필터링 기술

필터링(filtering) 기술은 NOTIFY를 통해 전달될 콘텐트에 대하여 가입자의 선호도(언제, 무엇을)가 무엇인지에 대한 규칙(rule)을 지정함으로써 프리젠티티의 상태 변화에 대한 모든 정보를 받지 않고 가입자가 지정한 필터 조건을 만족하는 정보만을 통지받도록 함으로써 가입자의 관심 외에 있는 불필요한 트래픽의 양을 줄일 수 있도록 한다. 필터는 가입자가 가입시 SUBSCRIBE 메시지의 바디에 포함시킬 수 있도록 하며 필터 포맷에 따라 정의된 유효한 XML 문서이다. 필터 포맷은 "application/simple-filter+ xml"으로 새로운 콘텐트 타입으로 등록되어 있다.

현재 IETF SIMPLE WG에서 필터링 기술은 "Functional Description of Event Notification Filtering"으로 제안된 상태이고, 필터 포맷은 "XML Based Format for Event Notification Filtering"으로 제안된 상태이다[17],[18].

일반적으로 각 패키지에서 가입자가 SUBSCRIBE 생성시 바디를 포함하지 않는 경우는 필터링을 하지 않는다는 의미이고, 바디가 포함되는 경우는 바디

• Filter Criteria Using <what> Element

```
<?xml version="1.0" encoding="UTF-8"?>
  <filter-set xmlns="urn:ietf:params:xml:ns:simple-filter">
    <ns-bindings>
      <ns-binding prefix="pidf" urn="urn:ietf:params:xml:ns:pidf"/>
      <ns-binding prefix="rpid" urn="urn:ietf:params:xml:ns:pidf:rpid-
      tuple"/>
    </ns-bindings>
    <filter id="123" uri="sip:presentity@example.com">
      <what>
        <include type="xpath">
            /pidf:presence/pidf:tuple[rpid:class="IM" or rpid:class="SMS"
            or rpid:class="MMS"]/pidf:status/pidf:basic
        </include>
    </what>
  </filter>
</filter-set>
```

• Filter Criteria Using <trigger> Element

```
<?xml version="1.0" encoding="UTF-8"?>
  <filter-set xmlns="urn:ietf:params:xml:ns:simple-filter">
    <ns-bindings>
      <ns-binding prefix="pidf" urn="urn:ietf:params:xml:ns:pidf"/>
    </ns-bindings>
    <filter id="123" uri="sip:presentity@example.com">
      <trigger>
        <changed from="CLOSED" to="OPEN">
          /pidf:presence/pidf:tuple/pidf:status/pidf:basic
        </changed>
      </trigger>
  </filter>
</filter-set>
```

(그림 9) 필터 문서의 예

정보에 포함된 필터를 반영하겠다는 것을 의미한다. 가입이 허용된 경우 그 다이얼로그가 유지되는 동안 필터가 유지되며, 기존 다이얼로그 내 새로운 SUBSCRIBE 메시지를 보냄으로써 기존 필터 대치, 정보 유지, 삭제, 기존 필터의 활성화(enabled)나 재활성화를 시킬 수 있다.

(그림 9)는 필터의 예이다. 첫번째 필터는 특정 조건을 만족하는 경우를 지정한 것이고, 두번째 필터는 특정 상태에 대한 지정된 변화가 발생할 때를 지정한 것이다.

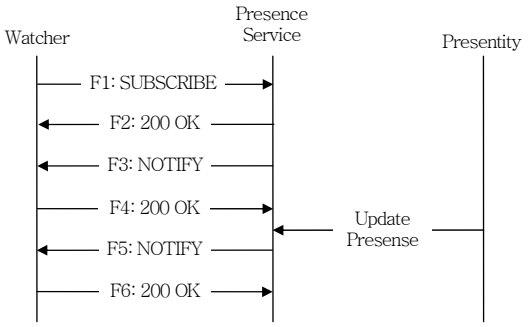
3. 부분 통지 기술

프레즌스 정보를 표현하는 최소 단위인 PIDF는 와처에게 전달될 튜플이라는 여러 조각의 데이터 모음으로 구성된다. PIDF를 기반으로 한 프레즌스 패키지에서는 프레즌스 정보의 소스인 프리젠티티에서 한 인스턴스의 변화로 인하여 PIDF의 일부 엘리먼트가 변경되더라도 PIDF 전체 문서가 전달된다. 부분 통지(partial notification) 방법은 프레즌스 서비스가 NOTIFY를 통해 와처에게 보낸 이전 프레즌스 정보와 비교해 튜플 단위의 변경된 부분만을 전달함으로써 트래픽 사이즈를 줄이고자 하는 개념이다.

현재 IETF SIMPLE WG에 “SIP Extensions for Partial Notification of Presence Information”이 제안된 상태이고 관련 포맷으로 “PIDF Extension for Partial Presence”로 제안되었고 현재 드래프트 상태이다[19],[20].

새로운 콘텐트 타입으로 ‘application/pidf-diff+xml’이 등록되어 있으며, 와처가 SUBSCRIBE 메시지 생성시 Accept 헤더에 해당 포맷과 q 파라미터를 통해 선호하는 포맷과 선호도를 지정할 수 있다.

(그림 10)은 와처가 프레즌스 서비스에게 부분통지를 선호한다는 가입 요청을 한 경우 프레즌스 서비스에서 이를 받아들여 즉시 F3을 통하여 통지를 하고, 프리젠티티에서 프레즌스 정보 변경이 발생했을 때 F5를 통하여 부분 통지를 하는 흐름을 보여준다. (그림 11)과 (그림 12)는 F3과 F5에서 전송되는 NOTIFY 메시지의 모습을 보여준다.



(그림 10) 가입 및 통지를 위한 메시지 처리 흐름

```
F3 NOTIFY sip:user@watcherhost.example.com SIP/2.0
Via: SIP/2.0/TCP server.example.com;branch=z9hG4bKna998sk
To: <sip:watcher@example.com>;tag=xfg9
From: <sip:resource@example.com>;tag=ffd2
Call-ID: 2010@watcherhost.example.com
Event: presence
Subscription-State: active;expires=3599
Max-Forwards: 70
CSeq: 8775 NOTIFY
Contact: <sip:server.example.com>
Content-Type: application/pidf-diff+ xml
Content-Length: [replace with real content length]

<?xml version="1.0" encoding="UTF-8"?>
<p:pidf-full xmlns="urn:ietf:params:xml:ns:pidf"
             xmlns:p="urn:ietf:params:xml:ns:pidf-diff"
             xmlns:r="urn:ietf:params:xml:ns:pidf:rpid"
                 entity="pres:someone@example.com" version="1">
<tuple id="sg89ae">
   <status>
      <basic>open</basic>
      <r:relationship>assistant</r:relationship> </status>
   <contact priority="0.8">tel:09012345678</contact> </tuple>
<tuple id="cg231jcr">
   <status>
      <basic>open</basic> </status>
   <contact priority="1.0">im:pep@example.com</contact> </tuple>
<tuple id="r1230d">
   <status>
      <basic>closed</basic>
      <r:activity>meeting</r:activity> </status>
   <r:homepage>http://example.com/~pep/</r:homepage>
   <r:icon>http://example.com/~pep/icon.gif</r:icon>
   <r:card>http://example.com/~pep/card.vcd</r:card>
   <contact priority="0.9">sip:pep@example.com</contact> </tuple>
   <note xml:lang="en">Full state presence document</note>
<r:person>
    <r:status>
        <r:activities> <r:on-the-phone/> <r:busy/> </r:activities>
       </r:status> </r:person>
<r:device id="urn:esn:600b40c7">
    <r:status>
        <c:devcaps> <c:mobility> <c:supported>
            <c:mobile/> </c:supported> </c:mobility> </c:devcaps>
       </r:status> </r:device>
</p:pidf-full>
```

(그림 11) Full State NOTIFY 메시지 예

```
F5 NOTIFY sip:user@watcherhost.example.com SIP/2.0
Via: SIP/2.0/TCP server.example.com;branch=z9hG4bKna998sl
To: <sip:watcher@example.com>;tag=xfg9
From: <sip:resource@example.com>;tag=ffd2
Call-ID: 2010@watcherhost.example.com
CSeq: 8776 NOTIFY
Event: presence
Subscription-State: active;expires=3543
Max-Forwards: 70
Contact: <sip:server.example.com>
Content-Type: application/pidf-diff+xml
Content-Length: [replace with real content length]

<?xml version="1.0" encoding="UTF-8"?>
<p:pidf-diff xmlns="urn:ietf:params:xml:ns:pidf"
             xmlns:p="urn:ietf:params:xml:ns:pidf-diff"
             xmlns:r="urn:ietf:params:xml:ns:pidf:rpid"
             entity="pres:someone@example.com" version="2">
<p:add sel="presence/note" pos="before">
<tuple id="ert4773">
   <status> <basic>open</basic> </status>
   <contact priority="0.4">mailto:pep@example.com</contact>
   <note xml:lang="en">This is a new tuple inserted
      between the last tuple and note element</note></tuple>
      </p:add>
<p:replace sel="*/tuple[@id='r1230d']/status/basic/text()">open
   </p:replace>
<p:remove sel="*/r:person/r:status/r:activities/r:busy"/>
<p:replace sel="*/tuple[@id='cg231jcr']/contact/@priority">0.7
   </p:replace>
</p:pidf-diff>
```

(그림 12) Partial State NOTIFY 메시지 예

4. 부분 등록 기술

앞에서 기술한 RFC3903 PUBLISH 방식은 일부 정보가 변경되었다 하더라도 바디에 모든 정보(full state)를 포함하여 전달한다. 즉, PIDF의 일부 엘리먼트가 변경되더라도 PIDF 전체 문서가 전달된다. 부분 등록(partial publication) 방법은 프리젠티티가 PUBLISH를 통해 프레즌스 서비스에게 보낸 이전 프레즌스 정보와 비교해 이후 변경된 프레즌스 문서의 일부만을 튜플단위로 등록함으로써 프리젠티티와 프레즌스 서비스 사이에 트래픽 사이즈를 줄이고자 하는 개념이다. 현재 IETF SIMPLE WG에 “Partial Publication of Presence Information”과 “PIDF Extension for Partial Presence”가 제안되어 현재 드래프트 상태이다[21].

부분 통지 방법은 초기에 PUBLISH 생성시 항상 모든 프레즌스 정보를 전달한다. 즉, Content-Type 헤더에 ‘application/pidf+ xml’이 지정되고, 바디에는 full state의 PIDF 문서가 포함된다. 그리고, 시작을 의미하기 위하여 버전 번호(version number)를 0으로 지정한다. 이후 프리젠티티에서 프레즌스 정보의 변경으로 PUBLISH 생성시 초기에 전달된 프레즌스 상태 정보와 비교해 변경된 부분만을 partial state로 표시하여 프레즌스 서비스에게 전달한다. 즉, Content-Type 헤더에 ‘application/pidf-diff+ xml’을 지정하고, 버전 번호는 이전에 보낸 것보다 1을 증가하여 지정한다.

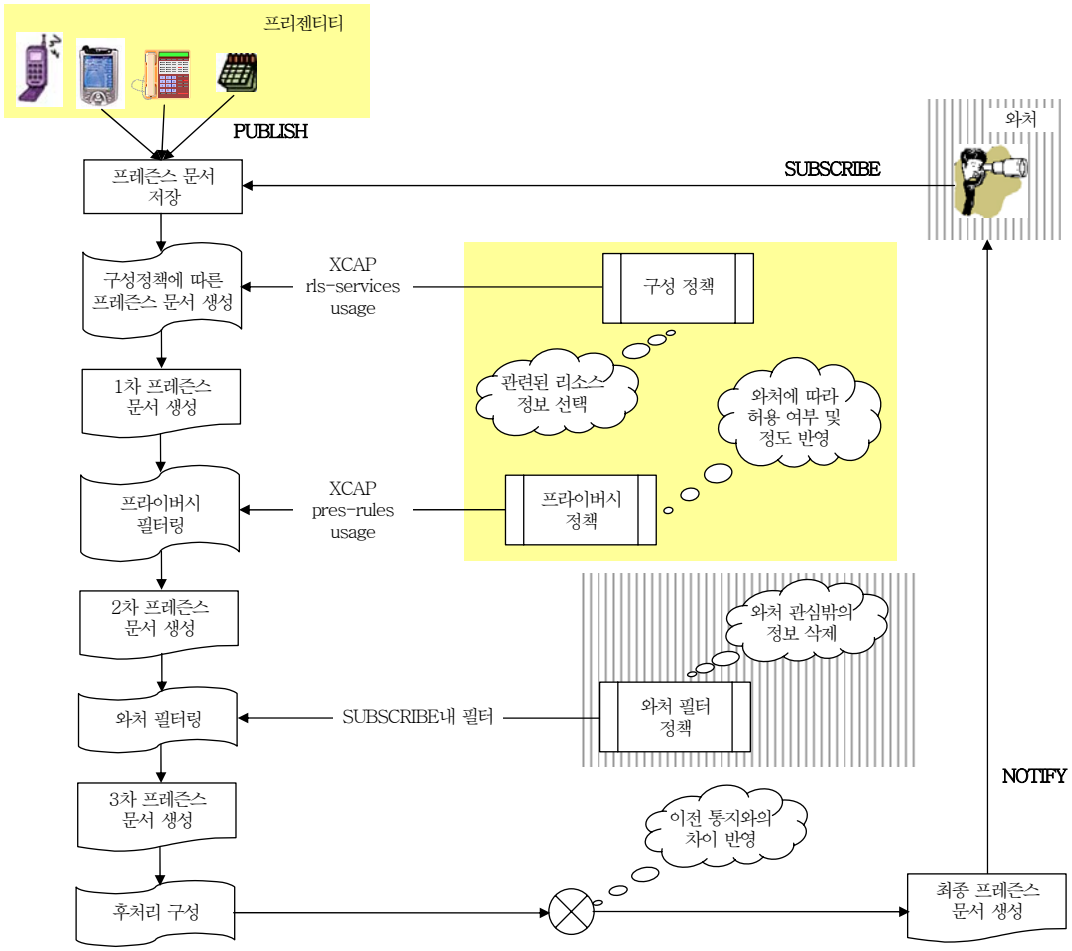
V. 프레즌스 정보 처리에 대한 전체 동작 흐름

(그림 13)은 프레즌스 소스인 프리젠티티에서 생성된 초기 프레즌스 문서가 와처에게 전달되는 형태의 최종 프레즌스 문서로 만들어지기 까지의 프레즌스 서비스에서의 처리 과정을 보여준다.

Ⅵ. 결론

본 논문에서는 SIP 기반 프레즌스 서비스를 위한 전체 표준화 동향을 간단히 살펴보았고, 특히 프레즌스 메시지 트래픽을 절감할 수 있는 방안에 대한 표준화 동향에 대하여 살펴보았다. 이러한 다양한 방법들이 전체 프레즌스 문서 처리 구조에서 어떻게 활용되는지도 간단히 살펴보았다.

SIP 기반 프레즌스 서비스는 SIP에서 정의한 컴포넌트를 재사용할 수 있으므로 관리 비용이나 새로운 통합된 서비스를 가능하게 하는 장점이 있다. 또한, 3GPP IMS의 Rel.6의 프레즌스 서비스에 활용되고 있으며, OMA의 PAG WG에서 관련 표준들이 사용되고 있다. 따라서, SIP 기반 프레즌스 프로토콜이 코어 기술로 활용되리라 예상되며 향후 SIP 기반 프레즌스 프로토콜을 기반으로 한 통합된 서비스 개발이 통신 가능성을 높일 수 있는 유용한 서비스가 되어 많은 사람들에게 각광을 받으리라 생각된다.



(그림 13) 프레즌스 정보 처리 흐름

약어 정리

3GPP	3rd Generation Partnership Project
EPA	Event Publication Agent
ESC	Event State Compositor
IETF	Internet Engineering Task Force
IMPP	Instant Messaging and Presence Protocol
IMS	IP Multimedia Subsystem
MIME	Multipurpose Internet Mail Extension
NGN	Next Generation Network
OMA	Open Mobile Alliance
PAG	Presence and Group
PIDF	Presence Information Data Format
RFC	Request For Comments
RLS	Resource List Server
RPID	Rich Presence: Extensions to the PIDF
SIMPLE	SIP for Instant Messaging and Presence Leveraging
SIP	Session Initiation Protocol
UA	User Agent
UAC	User Agent Client
UAS	User Agent Server
URI	Uniform Resource Identifier
XCAP	XML Configuration Access Protocol
XML	eXtensible Markup Language

참 고 문 헌

[1] http://www.ietf.org/html.charters/simple-charter.html

[2] RFC3261, SIP: Session Initiation Protocol, June 2002.
[3] http://www.ietf.org/html.charters/impp-charter.html
[4] RFC2778, Model for Presence and Instant Messaging, Feb. 2000.
[5] RFC3863, Presence Information Data Format(PIDF), Aug. 2004.
[6] draft-ietf-simple-presence-data-model-02, A Data Model for Presence, Feb. 21, 2005.
[7] draft-ietf-simple-rpid-05, RPID: Rich Presence: Extensions to the PIDF, Feb. 19, 2004.
[8] RFC3903, Session Initiation Protocol(SIP) Extension for Event State Publication, Oct. 2004.
[9] RFC3265, SIP-Specific Event Notification, June 2002.
[10] RFC3856, A Presence Event Package for the Session Initiation Protocol(SIP), Aug. 2004.
[11] RFC3857, A Watcher Information Event Template-Package for the Session Initiation Protocol(SIP), Aug. 2004.
[12] RFC3858, An Extensible Markup Language(XML) Based Format for Watcher Information, Aug. 2004
[13] P. Calhoun, J. Loughney, E. Guttman, G. Zorn, and J. Arkko, "Diameter Base Protocol," RFC 3588, Sep. 2003.
[14] draft-ietf-simple-presence-rules-02, Presence Authorization Rules, Feb. 21, 2005.
[15] draft-ietf-simple-event-list-07, A Session Initiation Protocol(SIP) Event Notification Extension for Resource Lists, Dec. 15, 2004.
[16] draft-ietf-simple-xcap-list-usage-05, Extensible Markup Language(XML) Formats for Representing Resource Lists, Feb. 7, 2005.
[17] draft-ietf-simple-event-filter-funct-05, Functional Description of Event Notification Filtering, Mar. 15, 2005.
[18] draft-ietf-simple-filter-format-05, XML Based Format for Event Notification Filtering, Mar. 15, 2005.
[19] draft-ietf-simple-partial-notify-05, SIP Extensions for Partial Notification of Presence Information, May 24, 2005.
[20] draft-ietf-simple-partial-pidf-format-04, PIDF Extension for Partial Presence, Feb. 2005.
[21] draft-ietf-simple-partial-publish-02, Partial Publication of Presence Information, Feb. 21, 2005.