특 집

VoIP를 위한 호처리 언어

안 상 현*, 이 종 화**, 박 선 옥***

● 목 차 ●

1. 서 론

IP 네트워크를 통해 음성전달을 가능케 하는 VoIP(Voice Over IP)가 소개되고 H.323, SIP와 같은 시그널링 프로토콜들이 제안되면서 텔레포니 서비스에 많은 변화를 가져왔다. 다양한 텔레포니 서비스를 서비스 공급자에게 요청할 필요 없이, 사용자가 직접 원하는 서비스를 명시하고 제어 할 수 있게 되었다. 또한 단순히 음성만을 전달하는 것이 아니라, 음성과 웹, 전자메일, 사용자의 상태정보 등을 통합한 다양한 서비스 제공이 가능하게 되었다 [2].

본 고에서는 호처리를 명시하고 제어하기 위해 사용될 수 있는 API를 SIP 기반으로 설명하며 구성은 다음과 같다. 2장에서는 호처리를 위한 네트워크 모델에 대해서 설명하고 3장에서는 여러 가지 호처리를 위한 API들 중에서 CGI, CPL(Call Processing Language), Servlet에 대하여 간략히 설명한다. 4장에서는 CPL의 구조와 태그, 확장에 대하여 상세히 설명한다.

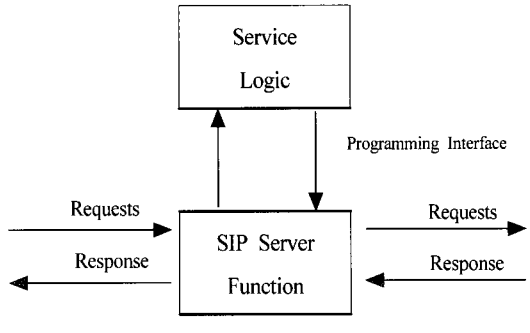
* 서울시립대학교 컴퓨터 · 통계학과 조교수
** 한국전자통신연구원 표준연구센터 선임연구원
*** 서울시립대학교 컴퓨터 · 통계학과 석사과정

2. 서비스 모델

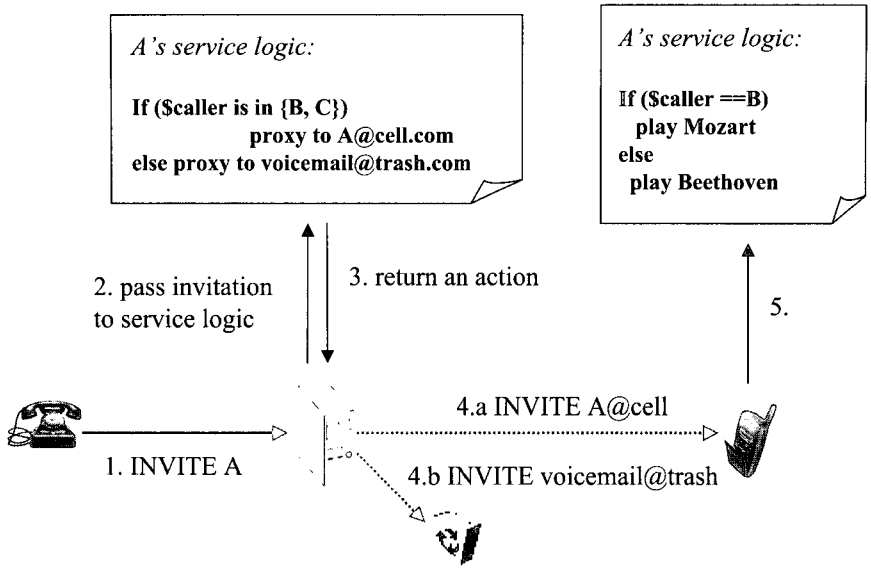
2.1 호처리를 위한 서비스 로직(Logic)

사용자는 네트워크 컴포넌트 중 네트워크 서버와 UAS(User Agent Server)에서 동작하게 될 호처리를 명시하는 서비스 로직을 생성한다.

생성된 서비스 로직은 (그림 1)에서와 같이 서버와 별개의 위치에 존재할 수도 있고 서버자체에 로직이 상주할 수도 있다. 전자의 경우에는 서버로 들어오는 콜 요청(Request)을 무조건 서비스 로직으로 넘겨주고 서비스 로직에서 호처리를 모두 수행하면, 인터페이스를 통해 결과 값을 서버에 되돌려 주는 방법을 사용하며, 후자의 경우에는 로직과 서



(그림 1) 서비스 로직의 모델



(그림 2) 서비스 로직의 예

버 사이에 별도의 인터페이스를 요하지 않는다 [4]. 서버에서 동작하게 될 서비스 로직을 말단 사용자가 직접 작성하기 때문에, CPL 서버는 로직이 서버로 업로드 될 때 로직이 정확하게 쓰여져 있는지를 체크하게 된다.

(그림 2)는 네트워크 서버와 UAS에 콜 요청이 들어왔을 때 취할 행동들을 구체적으로 명시한 예이다. 네트워크 서버에 사용자 A로의 콜 요청이 들어오면, 서버는 자신의 서비스 로직에 따라 호처리를 수행한다. 만약 콜러(Caller)가 B나 C인 경우에는 A의 셀룰러폰으로 콜을 연결하고 그렇지 않은 경우에는 음성메일로 연결시킨다. UAS에 존재하는 서비스 로직은 다음과 같은 동작을 수행한다. 만약 콜러가 B이면 모차르트 음악을, 그렇지 않은 경우에는 베토벤의 음악을 연주하도록 명시한다.

2.2 서비스 생성

서버에서의 호처리를 명시하는 스크립트 생성 방법에는 여러 가지가 있다. 사용할 API에 대한 지식을 습득한 사용자가 직접 스크립트를 작성할 수도 있고, GUI 도구를 사용하여 스크립트를 생성할 수도 있다. 현재, 스크립트 에디터와 같은 여러 가지 상용 제품들이 소개되어 있다.

3. 서비스 생성을 위한 API

사용자가 직접 자신에게 걸려오는 콜에 대하여 원하는 호처리를 명시하고 제어하기 위해서는 표준화된 API가 필요하다. IETF에서 제안한 가장 대표적인 API로는 CGI, CPL, Servlet 등이 있다.

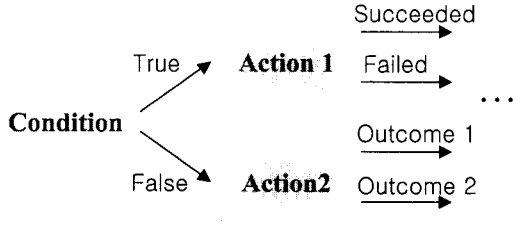
CGI는 인터넷 텔레포니 서비스를 처리하기에 충분하며 C나 Perl과 같은 언어로부터 독립적이다. 또한 여러 가지 목적을 위한 프로그래밍을 할 수 있다는 장점이 있으나 서버가 잘못된 스크립트에 쉽게 영향을 받는다는 단점을 가지고 있다. 반면, CPL은 콜을 어떻게 처리할 것인지를 명시하기 위한 목적만을 가진 언어이지만 서버에게 안전성을 제공해준다. 따라서 제3자에 의해서도 동적으로 로직 업로드가 가능하다. 마지막으로 Servlet은 자바 기반의 API로서 서버의 안전성도 일부 제공하고 여러 목적을 위해 프로그래밍 할 수 있는 일반성도 가진 언어이다[2, 4].

일반적으로 가장 적절한 서비스 생성 기법은 어떤 서비스를 명시하는지, 어디에서 실행되어 질 것인지에 따라 결정된다. 본 고에서는 이들 중 CPL에 대해서 상세히 알아본다.

3.1 CPL 개요

CPL은 XML 기반의 언어로서 네트워크 서버와 UAS에서 실행되어지며, 텍스트 기반의 태그로 구성된다. XML은 웹 상에서 데이터 교환을 위해 제안된 표준 언어로서, 사용자가 원하는 대로 문서의 구조를 정의할 수 있으므로 모든 형태의 데이터를 기술 가능하며 확장 또한 용이하다.

CPL은 기본적으로 조건(Condition)이 주어지고 해당하는 조건과 일치하면 그에 따른 행동(Action)을 취한다. (그림 3)은 CPL 결정 모델을 보여준다.



(그림 3) CPL 결정 모델

CPL로 구현 가능한 서비스들 중에서 간단한 예를 몇 가지 들어보면 다음과 같다. 실제로, CPL에서 제공하는 기본 태그들을 적절하게 조합함으로써 좀 더 다양하고 복잡한 서비스들을 표현할 수 있다.

- Call forward on busy/no answer
  콜러(Caller)의 상태에 따라 서로 다른 행동을 명시할 수 있다.
- Call forward on business hours
  콜이 처리되는 시간을 기반으로 서로 다른 곳으로 프록시 시킬 수 있다.
- Caller Selection
  콜을 한 콜러가 누구인지에 따라 서로 다른 행동을 명시할 수 있다.
- Intelligent user location with media knowledge
  콜 요청에 명시된 정보와 미디어 특성이 가장 잘 어울리는 곳으로 프록시 시킬 수 있다.

서비스를 기술한 CPL 스크립트는 스크립트가 실행될 서버까지 전송되어져야 한다. 전자메일이나 파일 업로드와 같은 방법으로 전송 가능하며 SIP REGISTER 메시지의 PAYLOAD 부분을 이용하여 전송하는 방법이 IETF에 의해 제안되어 있는 상태이다 [5].

4. CPL 구조와 태그[3]

CPL 스크립트는 크게 *Ancillary information*과 *Call processing action*이라는 두 가지 타입의 정보를 가진다. *Ancillary information*은 구문이 정의되어 있지는 않지만 확장을 위해 예약되어 있는 상태이다. 그리고 Call Processing Action은 콜 요청 시에 시그널링 서버가 취해야 할 결정과 행동을 명시한다. Call Processing Action은 콜이 들어올 때와 나갈 때에 취해야 할 행동을 명시하는 Top-level action과 스크립트 모듈화와 재사용을 위해 사용되는 Subaction으로 다시 나뉘어 진다.

Call Processing Action은 노드와 출력으로 구성되며, 각각의 노드와 출력은 XML 태그로 표현된다. 여기에서 말하는 노드와 출력은 (그림 3)에서의 Condition과 Action을 각각 의미한다. CPL 노드는 Switch, Location Modifiers, Signaling Action, Non-Signaling Action 등 4가지 종류가 있다.

4.1 Switches

모든 Switch들은 조건이 주어지고, 조건에 일치하는 출력이 실행된다, 조건에 맞지 않으면, OTHERWISE가 실행되며, OTHERWISE조차 없을 경우에는 Default Script Behavior가 실행된다. 현재

Address, String, Time, Priority 등 4가지 종류의 Switch가 제안되어 있는 상태이며 필요에 따라 확장 가능하다. (그림 4)는 XML 버전으로 표현한 간단한 Switch의 사용 예이다.

```
1 : <?xml version="1.0" ?>
2 : <!DOCTYPE cpl PUBLIC "-//IETF//DTD R FCxxxx
        CPL 1.0//EN" "cpl.dtd">
3 : <cpl>
4 :  <subaction id="voicemail">
5 :   <location url="sip:jones@voicemail.examp
        le.com">
6 :    <redirect />
7 :   </location>
8 :  </subaction>

9 :  <incoming>
10 :  <location url="sip:jones@phone.example. com">
11 :   <proxy timeout="8">
12 :    <busy>
13 :     <sub ref="voicemail" />
14 :    </busy>
15 :    <noanswer>
16 :     <address-switch field="origin">
17 :      <address is="sip:boss@exam ple.com">
18 :       <location url="tel:+19175551212 ">
19 :        <proxy />
20 :       </location>
21 :      </address>
22 :      <otherwise>
23 :       <sub ref="voicemail" />
24 :      </otherwise>
25 :     </address-switch>
26 :    </noanswer>
27 :   </proxy>
28 :  </location>
29 : </incoming>
30 : </cpl>
```

(그림 4) CPL 스크립트 예

Address Switch와 String Switch는 상당히 비슷한 구문을 가지며 콜 요청 메시지에 존재하는 주소 필드나 스트링 필드중의 하나를 기반으로 CPL 스크립트가 어떤 결정을 하도록 한다. (그림 4)의 16번째 줄부터 25번째 줄까지에 걸쳐 Address Switch가 사용된다. 비교하고자 하는 필드는 콜러의 주소이며 콜러의 주소가 "sip:boss@example.com"와 정확히 일치하면 명시된 전화번호로 연결시키며, 그렇지 않은 경우에는 음성메일로 연결시키도록 명시하고 있다.

Time Switch는 스크립트가 실행되는 시간이나 날짜를 기반으로 어떤 결정을 내리도록 한다. 구문 자체가 복잡해 보이기는 하지만 시간과 관련된 다양한 서비스가 표현 가능하다. 예를 들면 다음과 같다.

```
<time dtstart="19970105T083000" duration=
    "PT10M" freq="yearly" interval="2"
    bymonth="1" byday="SU">
```

interval="2"와 freq="yearly"는 격년을 의미하며, bymonty="1", byday="SU"는 1월의 매 일요일을 각각 의미한다. 따라서 위에 표현된 스크립트는 " 2년마다, 1월 매주 일요일 아침 8시 30분부터 아침 8시 40분까지"를 의미하게 된다.

Priority Switch도 마찬가지로 콜 요청 메시지에 명시된 우선순위에 따라 서로 다른 행동을 취하게 된다.

4.2 Location modifier

시그널링 서버는 등록된 Location Set을 기반으로 프록시를 시키게 되는데 이 Location Set에 새로운 Location을 추가시키거나 제거시키는 3가지 종류의 Location 노드가 정의되어 있다. 먼저, Explicit Location은 명시된 Location을 현재의 Location Set에 추가시킨다. <location> 태그를 사용하며 (그림 4)의 10번째 줄에서는 명시된 URL *"sip:jones@phone.example.com"*을 Location Set에 추가시킨다. 두 번째 노드로는 Location Lookup이 있으며

<lookup> 태그를 사용하고 source라는 필수 파라미터를 가진다. 다른 외부 소스로부터 Location을 얻어 와서 Location Set에 추가시키도록 하는 노드이다. source 파라미터는 주로 URL로 표현되며 non-URL source로는 현재 서버에 등록된 Location을 명시하는 registration이 정의되어 있다. 마지막으로 정의된 노드는 Location Set으로부터 특정 Location들을 제거할 수 있도록 하는 Location Removal이다. 사용하는 태그는 <remove-location>이다.

4.3 (Non-)Signaling Operation

시그널링 이벤트에 영향을 미치는 노드로는 Proxy, Redirect, Rejection 등이 있으며 (그림 4)의 6번째 줄과 11번째 줄에서 Redirect와 Proxy를 사용한 것이 나타나 있다.

또한, CPL은 텔레포니 시그널링 프로토콜에 영향을 미치지 않는 몇몇 노드들을 정의하고 있다. 서버가 콜러에게 CPL 스크립트의 상태를 전자메일을 통해 통보하기 위해 사용되는 *Mail* node와 서버가 콜에 대한 어떤 정보들을 로깅 할 수 있도록 하는데 사용하는 *Log* node 등이 있다.

4.4 Subaction

스크립트 재사용이나 모듈화를 위해 사용되며 마치 함수처럼 미리 정의해둔 Subacton을 불러 쓸 수 있다. (그림 10)의 4번째 줄부터 8번째 줄에서는 "sip:jones@voicemail.example.com"로 Redirect시키라는 Subaction을 정의하며 13번째 줄과 23번째 줄에서는 앞에서 정의해둔 Subaction을 각각 참조한다.

4.5 CPL 확장

CPL은 시그널링 콜을 어떻게 처리할 것인지에 대해 간단히 명시할 수 있는 강력한 도구이다. 그러나 사용자가 요구하는 서비스를 CPL로 해결하기 어렵거나 불가능한 경우, 예를 들어, 사용자가 어떤 비밀키를 가진 친구로부터의 콜만을 핸드폰으로 받길 원하는 경우, 인증 상태를 기반으로 콜을 처리해야 하지만, 현재 CPL 태그에는 이에 대한 정의를 하지 않고 있다. 따라서, 이러한 서비스를 제공하기 위해서는 CPL 확장이 필요하다. 이외에도 가능한 확장들은 많으며, 구체적인 스크립트 예를 하나 살펴보면 다음과 같다. 사용자가 특정한 Caller로부터의 콜에 대해 다른 벨소리를 원한다면, (그림 5)와 같은 스크립트를 사용할 수 있다. XML namespace "http://www.example.com/distinctive-ring"에는 새로운 노드 "Ring"이 명시된다.

```
<cpl xmlns="http://www.ietf.org/internet-drafts/draft-ietf-iptel-cpl-03.txt" xmlns:dr="http://www.example.com/distinctive-ring">
  <incoming>
    <address-switch field="origin">
      <address is="sip:boss@example.com">
        <dr:ring ringstyle="warble" />
      </address>
    </address-switch>
  </incoming>
</cpl>
```

(그림 5) 스크립트 예: Distinctive-Ring 확장

5. 결 론

인터넷 텔레포니는 전통적인 텔레포니 서비스와 다양한 인터넷 응용서비스를 통합 가능하게 하는 잠재성을 지니고 있다. 따라서 아직까지 존재하지 않았던 새로운 개념의 서비스가 가능해 질 것이다. 다양한 서비스를 다양한 사용자들에게 제공하기 위해서는, 서비스 공급자가 아닌 사용자에 의해서 서비스 기술이 되어져야 한다.

사용자들이 직접 자신에게 걸려오는 콜에 대한 호처리를 명시하기 위해서는 표준화된 API가 필요하다. 본 고에서는 여러 가지 API들 중에서 CPL이라는 호처리 언어의 구조와 태그, 확장 등에 대하

여 살펴보았다.

현재, VoIP의 주 흐름인 SIP 서비스를 제공하는 많은 제품들이 개발되어 있으며 CPL까지 지원하는 제품들도 소개되고 있다.

참고문헌

[1] J. Kuthan, "CPL Authentication and Database Access Extensions," Internet Draft, November, 2000.

[2] J. Lennox, H. Schulzrinne, "Call Processing Language Framework and Requirements," RFC 2824, May, 2000.

[3] J. Lennox, H. Schulzrinne, "CPL : A Language for User Control of Internet Telephony Services," Internet Draft, November, 2000.

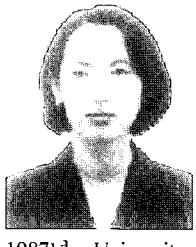
[4] J. Lennox, H. Schulzrinne, "Programming Internet Telephony Services," IEEE INTERNET COMPUTING, May · June, 1999.

[5] J. Lennox, H. Schulzrinne, "Transporting User Control Information in SIP REGISTER Payloads," Internet Draft, October, 2000.

저자약력

안 상 현

1986년 서울대학교 컴퓨터공학과 (공학사)
1988년 서울대학교 컴퓨터공학과 (공학석사)
1988년-1989년 데이콤 (사원)
1989년-1993년 University of Minnesota, Computer Science Dept (공학 박사)
1994년-1998년 세종대학교 컴퓨터학과 (전임강사/조교수)
1998년-현재 서울시립대학교 컴퓨터 · 통계학과 (조교수)
관심분야 : 인터넷, ATM, 라우팅, 멀티캐스트, QoS, VoIP 등
e-mail : ahn@venus.uos.ac.kr



이 종 화

1987년 University of Santiago, Chile 전산학과 졸업 (학사)
1990년 한양대학교 대학원 전자공학과 졸업(석사)
1990년-현재 한국전자통신연구원 표준연구센터 선임연구원
1996년 Technical University of Madrid, Spain 통신공학과 졸업 (공학박사)
관심분야 : 인터넷 텔레포니, 멀티미디어 통신 서비스, 사이버교육, 개방형 분산처리 시스템, 객체지향 설계 기법
e-mail : jhyi@pec.etri.re.kr

박 선 옥

1999년 서울시립대학교 컴퓨터 · 통계학과 (이학사)
2000년-현재 서울시립대학교 컴퓨터 · 통계학과 석사 과정
관심분야 : 신뢰적 멀티캐스트, VoIP, 네트워크 보안
e-mail : sopark95@venus.uos.ac.kr