主題

VoIP 호 처리 언어(CPL) 기술 동향

한국전자통신연구원 표준연구센터 통신프로토콜표준연구팀 이 종 화, 강 신 각

차 례

Ⅰ. 서론

인터넷 텔레포니 (VoIP) 분야에서 인터넷 전화 서비스를 비롯하여 프리젠스 및 인스턴트 메시징 서비스, 3rd party 콜 제어 서비스, 멀티미디어 화상회의 등 여러 유형의 응용 서비스들이 개발되고 있다. 인터넷 전화 서비스 경우 종단간 음성 통화뿐만 아니라 기존 유선 전화서비스와 같은 다양한 부가 서비스를 구현할 수 있다. 예를 들면, 호 전환, 무 응답 혹은 통화 중 착신 전환, 호 예약, 통화 중 대기, 회의 기능, 그룹 혹은 개별 당겨 받기 서비스, 호 필터링 서비스 등 여러 유형의 부가 서비스가 구현 가능하다.

인터넷상에서 이러한 부가 서비스들은 기존의 유선 전화 서비스와는 달리 사용자가 임의의 시간에 손쉬운 방법으로 자신이 직접 원하는 서비스를 등록하여 사용할 수 있어야 한다는 사용자들의 중요한 요구사항들을 만족시켜야 한다. 또한, 서비스 사업자들에게는 사용자들이 등록한 서비스를 실행시키는 환경이 SIP, H.323 등 하위 호 시그널링 프로토콜이나 사용자가 서비스를 등록하는 방식이나 환경에 독립적이어야 하는 요구사항이 만족되어야 한다. 이러한 요구사항을 만족시키기 위하여 초기에는 웹기반 혹은 자바기반의 서비스 생성, 등록, 통신 환경이 고려되다가 이들보다는 좀 더 손쉬운 에디팅 인터페이스를 제공하고, 여러 종류의 수송계층 프로토콜 사용이 가능하며, 처리 시간이 비교적 빠른 XML기반의 호 처리 언어가 탄생하게 되었다. 호 처리 언어란 인터넷 텔레포니 서비스를 기술하고 제어할 수 있는 툴을 의미한다. 즉, 위에서 언급한 부가서비스를 사용자가 선택·기술·등록하여 자신이 수신하는 호를 제어할 수 있도록 하는 도구로 활용될 수 있음을 의미한다.

호처리 언어에 대한 표준화는 IETF IPTEL (IP Telephony) 워킹그룹이 구성된 1999년도에 시작되었고, CPL 표준문서가 00버전에서부터 06버전까지 발표되어 마지막 버전이 2002년 2월에 정식 표준으로 채택되었다. 호처리 언어기반 서비스 모델은 하위 수송계층 프로토콜로서 현재 VoIP에서 사용되고

있는 SIP와 H.323 프로토콜을 고려하고 있으며, CPL이 제공하는 기능에 대해 이들 프로토콜과의 매핑도 함께 정의하고 있다.

현재 SIP 프로토콜 기반 호처리 언어를 구현한 몇몇 제품이 공개되고 있으며, 이를 구현한 주요 업체로는 Dynamicsoft, Indigo Software, Ubiquity, Netcentrex 등이 있다. SIP프로토콜을 오픈 소스로 공개하고 있는 Vovida에서도 CPL를 지원하는 Proxy서버를 구현하여 오픈 소스로 공개하고 있다.

본 논문에서는 먼저 CPL 관련 표준화 활동에 대해 소개하고, CPL 표준문서인 06버전에 정의되어 있는 CPL 스크립트의 구조, 기능, 예제 등을 기술한다. 또한 현재까지 구현된 주요 CPL 제품의 기능과 구조를 소개한다.

Ⅱ. 표준화 현황

CPL에 대한 표준화는 IETF에서 IPTEL 워킹그룹[1]이 구성된 1997년 11월에 시작되었으며, IPTEL 워킹그룹은 CPL이외에도 TRIP (Telephony Routing over IP)에 관한 표준화를 추진하고 있다.

CPL관련 표준화는 CPL을 지원하는 프레임워크와 호처리를 위한 서비스 모듈에 대한 요구사항 분석이 먼저 수행되어, 현재 이에 대한 표준이 정의되어 있다[2]. 이 표준에서는 CPL을 지원하는 네트워크 구성요소들을 정의하고, 특히 CPL을 지원하는 엔드 시스템의 기능 그리고 네트워크 구성요소간의 상호동작을 기술하고 있다.

CPL과 관련된 다른 기술적 이슈는 사용자가 선택한 서비스 로직을 어떻게 실어 서버에 전달하느냐를 다루는 업로드 방식이다. 이를 위해 제안된 방식으로는 SIP프로토콜의 REGISTER 메시지를 사용하여 Registrar에 등록하는 방식[3]으로서, SIP의 주요 멤버로 활동하고 있는 콜롬비아대학의 Jonathan Lennox와 Henning Schulzrinne이 제안하였다. [3]에서는 SIP REGISTER 메시지[4]를 이용하여 서비스 로직 (즉, CPL 스크립트)을 전달하기 위하여 SIP 에서 정의되어 있는 Content-Disposition 헤더를 수정·확장하고, Accept-Disposition과 If-Unmodified-Since라는 새로운 헤더를 정의하고 있다. 또한, 스크립트 전송과 삭제에 대하여 User Agent와 Registrar의 동작 기법을 정의하고 있다. 이 개인 드래프트는 2000년 10월에 제안되었으나 새로운 헤더 추가와 동작의 복잡성이라는 이유로 호응을 받지 못하고 2001년 4월에 종료되었으며, 이와 관련된 다른 표준은 아직까지 제안되지 않고 있지 않는 상황이다.

CPL 표준은 1999년 4월에 첫 드래프트(draft-ietf- iptel-cpl-00)가 발표되었고, 이것이 수정 보완되는 작업을 거쳐 06버전의 드래프트가 2002년 1월 정식적으로 발표되었다[5]. IPTEL 워킹그룹은 CPL에 대한 드래프트 문서를 정식 RFC로 채택되게 하기 위하여 이를 IETF IESG 그룹에 제출하였으며, 2002년 2월 초 IPTEL 워킹그룹은 메일링 리스트를 통해 CPL 06버전이 RFC로 채택되었음을 공지하였다.

Ⅲ. CPL 구조 및 기능

1. 개요

[4]에서 정의하고 있는 CPL 표준 문서는 크게 CPL 스크립트의 구조, 신텍스, CPL이 제공하는 기능, CPL 기반 서비스 예제 그리고 호 시그널링 프로토콜로서 H.323을 사용하는 경우의 사용법 등으로 구성되어 있다. 이 중 CPL이 제공하는 기능은 세부적으로 Switches, Location Modifiers, Signalling Operations 그리고 Non-signalling Operations로 구분하고, 각각에 대해 조건, 주어진

조건에 맞는지에 대한 결과에 따라 수행해야 하는 동작 그리고 주어질 수 있는 파라메터에 대해 정의하고 있다. 또한, 이러한 기능들이 SIP 프로토콜을 사용하는 경우 매핑 방법도 함께 정의하고 있다.

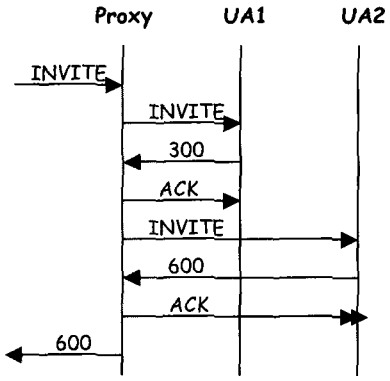


그림 1. CPL의 생명주기

CPL은 XML을 기반으로 하는 언어로서 텍스트 기반의 태그로 구성되며, 기본적으로 XML로 구성되는 문서들이 갖는 장점들 (예를 들면 쉬운 편집, HTTP, SIP, FTP 등 여러 유형으로의 전달이 가능한 점, 사용자 고유 목적을 위한 확장 용이성, 비교적 적은 사이즈, 빠른 수행 동작)을 제공한다. 구조적인 면에서 CPL은 하나의 XML 문서 형태를 취하며, [4]에서 정의하고 있는 CPL 기능들을 적절히 조합함으로써 다양한 VoIP 서비스를 표현할 수 있다.

CPL의 실행은 SIP를 이용하는 경우 Proxy서버나 SIP 응용 서버에서 그리고 H.323을 이용하는 경우에는 게이트키퍼에서 이루어 질 수 있다. 이것은 각 서버에서 CPL을 이해하고 실행시킬 수 있는 실행 환경 (Execution Environment)을 지원함을 의미한다. Proxy서버에서 CPL 스크립트를 실행하는 경우를 고려하는 경우, 한 스크립트의 생명주기(lifecycle)는 CPL 스크립트가 Invite 메시지를 통해 Proxy 서버에 도착한 시점에서 시작되어 Proxy에 의해 Invite 메시지가 처리되고 이에 대한 최종 응답이 전달될 때 종료된다. (그림 1)에서 사각 처리된 부분이 Invite 트랜잭션에 해당하는 CPL 생명주기의 예를 보여주고 있다.

2. 구조

하나의 CPL 스크립트는 조건(Condition)과 행동(Action)으로 구성되는데, 이것의 의미는 주어진 조건과 일치 하는지를 판단하여 일치하는 경우와 그렇지 않은 경우에 해당하는 행동을 정의하게 된다. CPL 스크립트는 노드와 연결 링크를 갖는 트리 구조를 갖는다. 여기에서 노드는 조건에 따라 수행해야 할 행동을 정의하고 연결 링크는 행동에 대한 결과를 정의한다[6]. CPL은 크게 〈incoming〉, 〈outgoing〉 그리고 〈subaction〉으로 구성될 수 있는데, 〈incoming〉과 〈outgoing〉은 전달 받은

```
<?xml version=1.0 ?>
<!DOCTYPE cpl PUBLIC -//IETF//DTD RFCXXXX CPL 1.0//EN cpl.dtd>

<cpl>
    <subaction id=voicemail>
        ----- Scripts -------
    </subaction>

    <subaction id=mail>
        ----- Scripts -------
    </subaction>

    <incoming>
        ----- Scripts -------
    </incoming>

    <outgoing>
        ----- Scripts -------
    </outgoing>
</cpl>
```

그림 2. CPL 스크립트의 구조

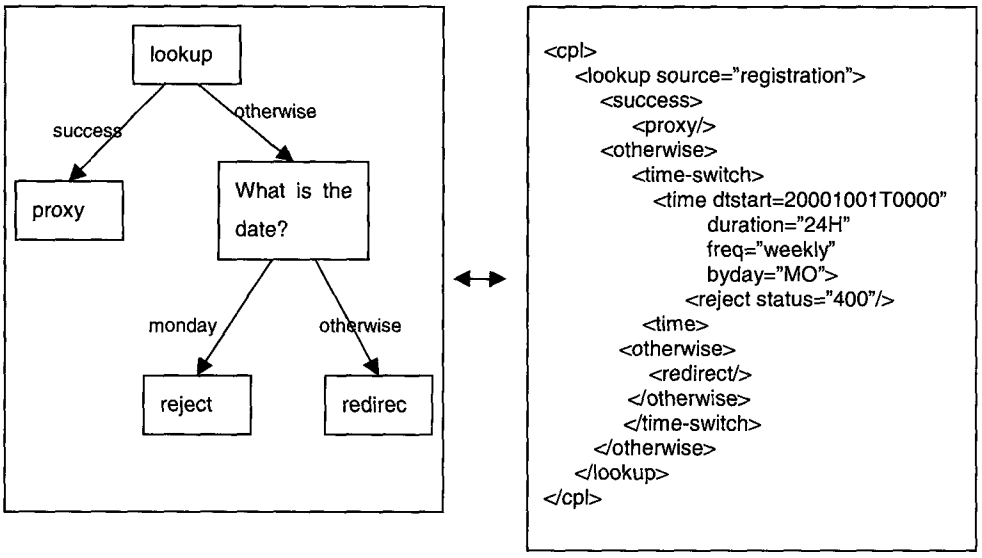


그림 3. CPL 스크립트의 예

호에 대해 혹은 수신할 호에 대해 수행하는 모듈에 해당한다. 그리고 모든 호에 대해 모듈화와 재사용을 위해 서브 루틴을 정의할 수 있는데 이를 위해 〈subaction〉을 이용한다.

(그림 2)는 CPL 스크립트의 전체 프레임을 보여주고 있다. 맨 처음에 XML 버전과 포함시켜야 하는 CPL DTD(Documment Type Definition) 화일에 대해 정의하는 것으로 시작한다. 그 다음 〈cpl〉 태그를 이용하여 스크립트 부분을 정의하게 되는데, 〈cpl〉 태그안에는 하나 이상의 〈subaction〉과 0 혹은 하나의 〈incoming〉과 〈outgoing〉이 올 수 있다.

(그림 3)은 CPL 노드와 연결 링크를 이용하여 간단한 CPL기반 서비스의 예를 트리 구조로 보여주고 있으며, 이 서비스에 해당하는 스크립트 표기도 함께 보여주고 있다.

CPL 표준 문서에서는 하위 네트워크 구성요소와 CPL 실행 환경과의 상호 동작을 정의하고 있지는 않지만, CPL하위 호 시그널링 프로토콜을 SIP를 사용하는 것을 고려하는 경우 다음과 같이 전체 동작 시나리오를 고려할 수 있다. 먼저, 사용자가 원하는 서비스에 대한 CPL 스크립트는 앞에서 언급한 방식으로 등록되고, 이 후 사용자는 호 설정을 위해 Invite 메시지를 User Agent를 통해 Proxy서버로 전달하게 된다. Proxy 서버는 전달 받은 Invite 메시지를 처리하기 전에 CPL 실행 환경 혹은 CPL 서버에게 해당 호에 대해 이미 등록된 서비스 로직이 있는가를 요청하게 된다. 이러한 요청을 받은 CPL 서버는 해당 호에 대한 CPL 스크립트를 업로드하여 〈incoming〉 혹은 〈outgoing〉 태그에 대해 주어진 조건에 일치하는 행동을 결정하기 위해 프로세싱을

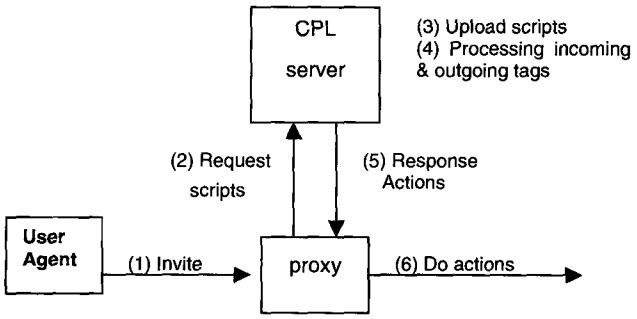


그림 4. Proxy 서버와 CPL 서버와의 상호동작

하게 된다. 프로세싱 결과로서 얻어진 행동은 Proxy 서버에게로 전달되어 지고, Proxy 서버는 이를 수행하게 된다. 만약 등록된 서비스 로직이 없는 경우 Proxy서버는 기존의 자신의 동작대로 처리하게 된다(그림 4 참조).

3. 기능

CPL 표준 06버전에서는 CPL 기능을 5 그룹으로 구분하여 (그림 5)와 같이 정의하고 있다. CPL 스크립트는 Ancillary information과 Call processing action 으로 구분되는데, 첫번째에 해당하는 구문은 현재 정의되어 있지 않고 향후 확장을 위해 예약되어 있는 상태이며, 앞에서 기술한 모든 CPL 태그들은 두번째 정보에 속한다.(그림 5)에서 나타나는 각 CPL 태그에 대한 기능 정의는 다음과 같다.

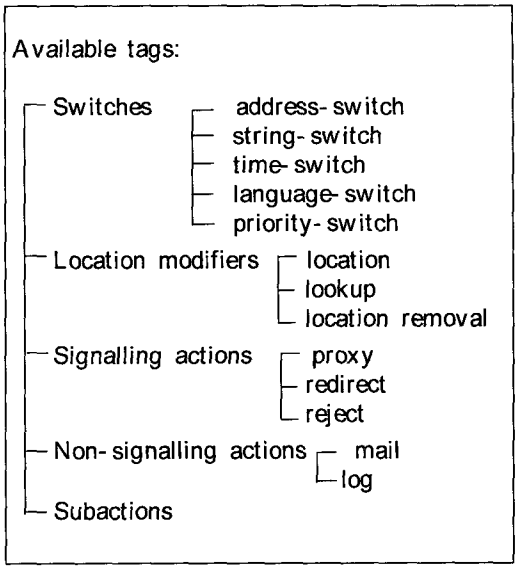


그림 5. CPL Tags

(1) Switches : 현재 5개의 세부 스위치 태그가 정의되어 있는데, 공통적으로 X-switch라는 구문을 사용한다. 이 때 X는 address, string, language, time, priority 로 대치되며, 주어지는 조건의 대상을 의미한다. 예를 들면, CPL 서버는 Proxy로부터 요청 받은 호의 address에 대해 조건이 일치되는지 분석하여 수행해야 될 행동을 결정하게 된다. 이 때 address 가 갖을 수 있는 세부 파라메터에 대한 조건이 정의된 경우 세부 파라메터가 조건의 대상이 되어 프로세싱된다.

세부 노드	output	parameters	
address-switch	address	field	origin, destination, original-destination
		subfield	address-type, user, host, port, tel, display, password, alias-type
string-switch	string	field	subject, organization, user-agent, display
languge-switch	language	none	
time-switch	time	tzid, tzurl	
		dtstart, dtend, duration, freq, interval, until, count, bysecond, byminute, byhour, byday, bymonthday, byyearday, byweekno, bymonth, wkst, bysetpos	
priority-switch	priority	less, greater, equal	

(2) Location Modifiers: 주어진 위치(location)를 기존의 위치 정보 리스트에 추가, 삭제 혹은 검색하는 기능을 수행하게 된다. 정의된 파라메터를 이용하여 세부적으로 조건을 정의할 수 있다

세부 노드	output	parameters
explicit location	none	url, priority, clear
location lookup	success, notfound, failure	source, timeout, use, ignore, clear
location removal	none	location, param, value

(3) Signalling actions: 결정된 CPL 행동이 시그널링 동작에 해당하여 하위 시그널링 프로토콜에 영향을 주게 된다. 시그널링 동작은 수정된 주소로 "proxy" 하거나 "redirect" 가 되는 경우나 호 설정에 대해 "reject"를 수행할 수 있다.

세부 노드	output	parameters
proxy	busy, noanswer, redirectionm failure, default	timeout, recurse, ordering
redirect	none	permanent
reject	none	status, reason

(4) Non-signalling actions: Signalling actions과는 달리 하위 시그널링 프로토콜에 영향을 주지 않는 동작에 해당하며, 현재 "Mail"과 "log" 태그가 정의되어 있다

세부 노드	output	parameters
Mail	none	url
Log	none	name, comment

(5) Subactions: 이미 정의한 스크립트의 재사용과 모듈화를 위해 사용된다. 각 subaction에 고유 id가 할당되고, 다른 태그안에서는 "sub ref=" 를 이용하여 subaction을 불러 사용할 수 있다

세부 노드	parameters
subaction	Id
Sub	Ref

Ⅳ. 제품 동향

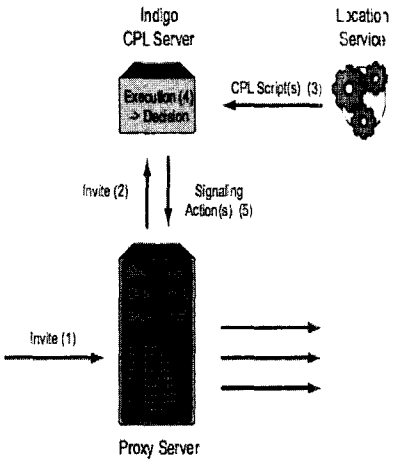
현재 CPL를 구현한 제품을 릴리즈하고 있는 업체로는 Indigo software, Dynamicsoft, Ubiquity software corporation, Netcentrix 등이 있으며, Vovida Vocal에서는 Feature Server라는 명칭으로 CPL 기능에 대한 오픈 소스를 공개하고 있다. 이들 대부분은 하위 시그널링 프로토콜로서 SIP를 사용하고 있고, Indigo software를 제외한 나머지 업체들은 SIP Proxy server라는 명칭하에 Proxy 기능에 CPL 을 지원하는 실행 환경이 포함되어 개발.공개되고 있으며, Indigo에서는 CPL Server version 2.1을 별도로 발매하고 있으나 이 제품은 Indigo의 Proxy server상에서만 동작하도록 개발되어 있다.

1. Indigo software [7]

CPL 관련 제품으로 CPL editor version 2.1과 CPL Server version 2.1이 있으며, CPL을 지원하는 SIP Proxy/Redirect Server 그리고 Registrar가 발표되어 있다. CPL Editor는 사용

자가 직접 그래픽 인터페이스를 통해 인터넷 전화 서비스를 기술하고 제어할 수 있는 기능을 제공하고 있으며, 기본적으로 송신자/수신자의 주소와 이름, 주제(subject), 송신자의 소속기관명, 호에 대한 선호도 및 시간을 제어할 수 있는 서비스를 지원한다. CPL Server는 사용자가 등록한 스크립트를 업로드하는 기능, 스크립트에 대한 분석, 실행 그리고 Proxy에게 결정된 행동을 전달하여 수행하도록 하는 기능이 포함되어 있다. 사용자의 스크립트는 [3]에서 정의한 방식으로 Registrar에 등록된다. CPL Editor나 Server는 Unix, window 환경에서 Java 2 version 1.2를 사용하여 개발되었다.

(그림 6)에서 CPL Editor와 Server의 동작 구조를 보여주고 있다.

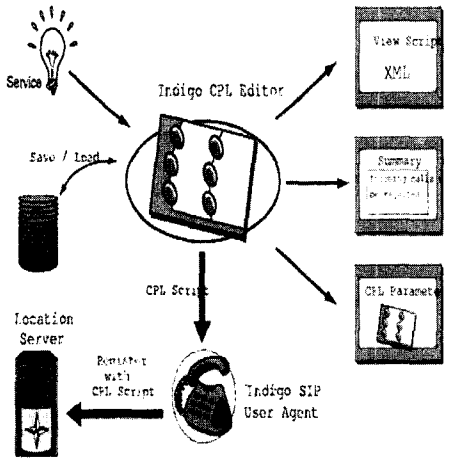


(b) CPL Server

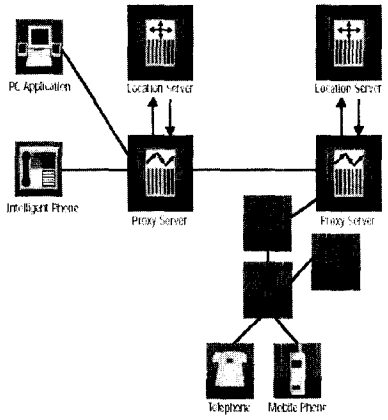
그림 6. Indigo CPL Products

2. Dynamicsoft [8]

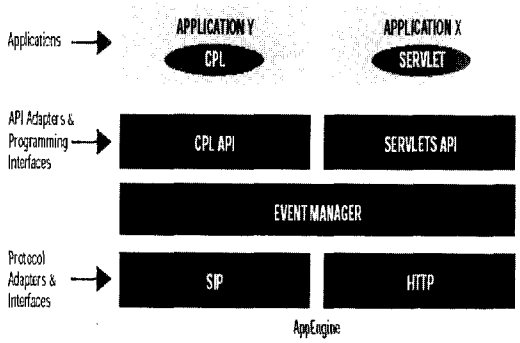
CPL 관련 제품으로는 CPL를 지원하는 Proxy server와 다양한 서비스를 개발할 수 있는 툴을 지원하는 AppEngine version 2.1을 들 수 있다(그림 7참조). 먼저, Dynamic SIP Proxy Server 5.1은 SIP 프로토콜의 표준인 RFC 2543을 따라 SUN Solaris 2.8과 window 환경에서 JVM 1.2를 이용하여 개발되었다. 이 Proxy 5.1에 CPL를 처리하는 실행 모듈이 포함되어 하나의 제품으로 공개되고 있다. AppEngine은 음성, 비디오, 웹, 전자 메일, 프리젠스 및 인스턴트 메시지 서비스를 혼합하여 여러 유형의 새로운 서비스를 개발할 수 있는 플랫폼과 APIs를 제공한다. 수송 프로토콜로서는 SIP를 비롯하여 HTTP, SMTP를 지원하며 CPL API와 Servlets API를 지원한다.



(a) CPL Editor



(a) Dynamic SIP Proxy Server 5.1



(b) Dynamic AppEngine 2.1

그림 7. Dymamic Products

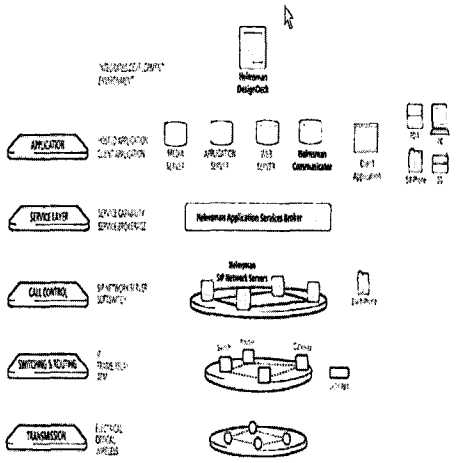


그림 8. Ubiquity Appication Services Platform

3. Ubiquity software corporation [9]

Ubiquity에서는 Helmsman Application Services Broker 라는 명칭으로 제품을 공개하고 있는데, 이것은 여러 개발 툴을 이용하여 서비스를 개발할 수 있도록 하는 서비스 모듈과 데이터를 송수신하는 전송 모듈로 구성된다. 서비스 모듈에는 CPL을 비롯하여 프리젠스, 인스턴트 메시징, 웹, 3rd party call control 등의 기능을 이용하여 다양한 서비스를 개발할 수 있는 플랫폼을 제공한다. 전송 모듈은 기본적으로 SIP 네트워크를 사용하여 호를 설정하고 제어할 수 있도록 한다. CPL 스크립트의 등록 방식은 다른 제품들과 동일하게 Register 메시지를 이용하고, CPL 스크립트는Application Services Broker의 한 구성요소인 CPL Storage에 저장 관리된다.

(그림 8)은 Ubuquity Application Services 플랫폼의 전체 구조를 보여주고 있다.

4. Vovida [10]

Vovida에서는 기존의 Proxy 기능을 수행하는 서버를 Marshal Server 라 부르고, 이 서버에 CPL 기능을 지원하는 서버를 Feature Server 라 칭하고 있다. 현재 공고되어 있는 Feature Server는 버전이 1.3.0으로서 CPL 표준문서 01버전에 따라 C++를 이용하여 Linux 7.2, Solaris 8상에서 개발되었다. Vovida가 지원하는 CPL기반 서비스는 900/976 blocking, Caller ID Blocking, Call Return, Call Screening, Call Forward No Answer/Busy 등이 있다. (그림 9)에서 Vovida Vocal 시스템의 전체 구조를 보여주고 있다.

5. 기타 소프트웨어

CPL 표준에서 정의되어 있는 태그 중에 Time-switch 기능을 구현한 소프트웨어가 Cal-Code라는 이름으로 공개되어 있는데[11], 이것은 RFC 2445 (Internet Calendaring and Scheduling Core Object Specification)의 일부분에 해당한다. Cal-Code는 콜롬비아 대학교에 의해 Java로 구현되었다.

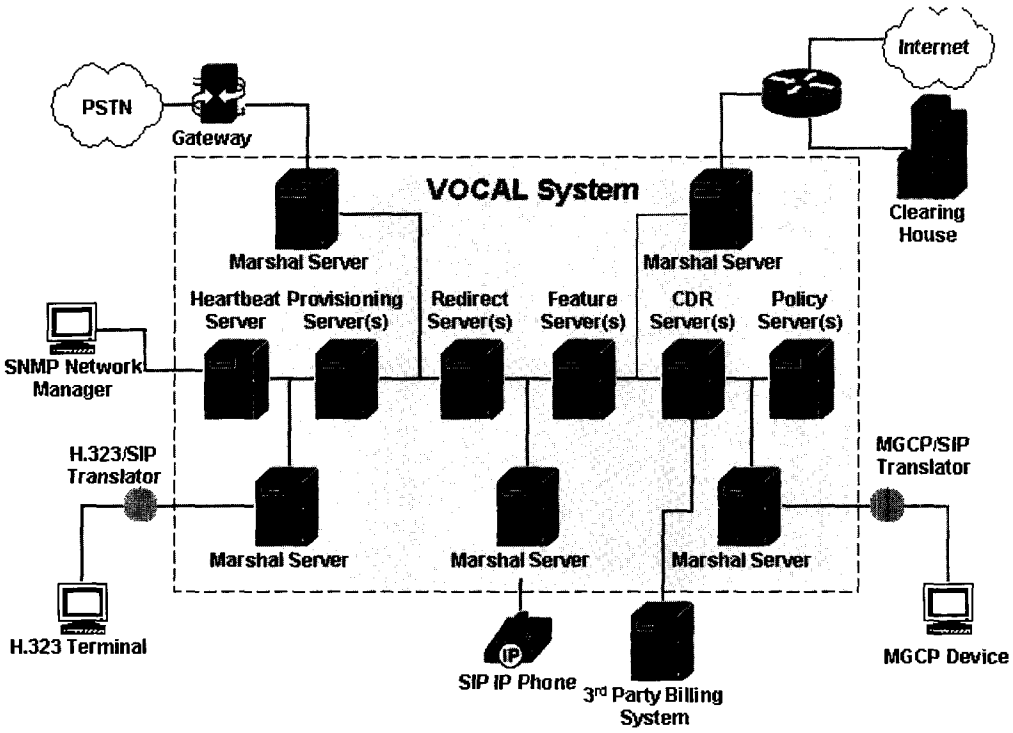


그림 9. Vovida Vocal Architecture

V. 결론

본 논문에서는 VoIP 분야에서 다양한 응용 서비스를 개발하고자 할 때 사용할 수 있는 호처리 언어에 대해 기술하였다. 호처리 언어는 현재 VoIP에서 가장 대표적인 응용의 하나인 인터넷 전화 서비스와 연계되어 여러가지의 부가 서비스를 구현할 수 있는 도구에 해당한다. 기존의 유선 전화서비스에서 제공받고 있는 호 전환, 호 예약, 삼자 통화, 통화중 대기 등의 서비스가 인터넷상에서 쉽게 개발될 수 있다. 무엇보다도 사용자가 직접 자신이 원하는 조건에 맞추어 (예를 들면 수신자명, 시간, 선호도 등에 대한 조건) 선택하여 서비스를 제공받을 수 있게 한다는 점에서 CPL의 장점을 찾을 수 있다.

CPL에 대한 표준문서도 이제 정식 표준으로 채택되어 실제 VoIP 서비스 구현시 CPL의 활용도를 좀 더 높일 수 있으리라 전망되며, 더 나아가서는 음성뿐만 아니라 기존 혹은 차세대 인터넷 응용 서비스와의 통합을 통해 경쟁력이 있는 새로운 개념의 서비스들의 구현을 촉진시킬 수 있지 않을 까 기대해 본다.

참고문헌

[1] IETF iptel WG, http://www.ietf.org/html. charters/iptel-charter.html

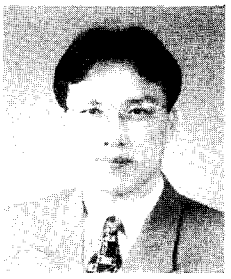
[2] IETF RFC 2824, Call Processing Language Framework and Requirements , 05 2000.

[3] J. Lennox, H. Schulzrinne, IETF draft-lennox-sip-reg-payload-01.ps, "Transporting User Control Information in SIP REGISTER Payloads", 10 2000.

[4] IETF RFC 2543, "SIP: Session Initiation Protocol", 03 1999.
[5] IETF draft-ietf-iptel-cpl-06.ps, "CPL: A Language for User Control of Internet Telephony Services", 01 2002 .
[6] 상현, 이종화, VoIP를 위한 호처리 언어 , 한국정보처리학회지 제8권 제2호, 03 2001.
[7] http://www.indigosw.com/
[8] http://www.dynamicsoft.com/
[9] http://www.ubiquity.net/
[10] http://www.vovida.org/
[11] http://www.cs.columbia.edu/~lennox/Cal-Code/

이 종 화

1990년 2월 한양대학교 대학원 전자공학과 졸업(석사), 1990년 2월 ~ 현재 한국전자통신연구원 표준연구센터 선임연구원, 1996년 11월 Technical University of Madrid, Spain 통신공학과 졸업 (공학박사), 〈관심분야〉 VoIP, 인터넷 응용, 멀티미디어 통신, 개방형 분산처리 시스템, 객체지향 설계 기법



강 신 각

1984년 충남대학교 전자공학과 (학사), 1998년 충남대학교 전자공학과 (박사), 1984년 -현재 한국전자통신연구원 통신프로토콜표준연구팀 팀장/책임연구원, 1995년 정보통신기술사, 1997년-현재 ITU-T SG 7 Rapporteur, 2000년-현재 인터넷 텔레포니 포럼 부의장/운영위원장 〈관심분야〉 멀티캐스트 통신, VoIP, 인터넷보안