Call Processing Language (CPL) Based Service Configuration System

Mahavir D Karnavat
Shivaji Hogale

Call services are getting more popular these days as new call processing technologies like VOIP are introduced to the market and Internet is being accepted by masses. It is becoming difficult for the equipment vendors to support all possible call services and allow the end users to subscribe them independently. This is because the number of possible combinations of the call services is very high. In order to facilitate this, there is a need to support the basic call features (like call forwarding, sending SMS/email etc) by the vendors and allow the service providers to enable the end-users to personalize their call services. This leads to another advantage of saving the cost of manufacturing as the manufacturer need not worry about the possible call services, rather, the manufacturer can just implement the basic call features (like providing JCC APIs) and leave the customization to the service provider / end-user. This white paper proposes an architecture that allows the end user to personalize their call services. It uses the call processing language (CPL), a lightweight extensible language, which is being standardized by IETF.
Table of Contents

INTRODUCTION ........................................................................................................... 3
1 DRIVERS FOR CPL BASED SERVICE
   CONFIGURATION SYSTEM .................................................................................. 4
   1.1 Problem at hand ....................................................................................... 4
   1.2 What is CPL .............................................................................................. 4
   1.3 Features of CPL ...................................................................................... 5
   1.4 Applicability ........................................................................................... 6
   1.5 Benefits of using CPL ............................................................................ 7
   1.6 Open Issues ............................................................................................. 7
2 PROPOSED SYSTEM ARCHITECTURE ......................................................... 8
   2.1 Service configuration user Interface ...................................................... 8
   2.2 Service Invocation System ...................................................................... 10
3 ALTERNATE TECHNOLOGIES AVAILABLE IN THIS SPACE ........ 13
   3.1 Other Technologies ............................................................................... 13
   3.2 Pros and cons of using CPL .................................................................. 14
4 APPLICATIONS ................................................................................................. 15
   4.1 Example call services .......................................................................... 16
   4.2 Usage scenarios ..................................................................................... 16
CONCLUSION ............................................................................................................. 17
ACRONYMS AND REFERENCES .............................................................................. 17
REFERENCES .......................................................................................................... 18
ABOUT THE AUTHORS .......................................................................................... 18
ABOUT WIPRO TECHNOLOGIES .......................................................................... 19
Introduction

Recently, several protocols have been created to allow telephone calls over IP networks, notably SIP [1] and H.323 [2]. These emerging standards have opened up the possibility of a broad and dramatic decentralization of the provisioning of telephone services so they can be under the user's control.

Traditionally, network-based services are created only by the service providers. Service creation typically involved using proprietary or restricted tools, and there was little range for customization or enhancement by end users. In the Internet environment, however, this changes. Global connectivity and open protocols allow end users or third parties to design and implement new or customized services, and to deploy and modify their services dynamically without requiring a service provider to act as an intermediary.

IETF IPTEL work group has proposed an architecture in which network devices respond to call signaling events by triggering user-created programs written in a simple, static, non-expressively-complete language. This language is called Call Processing Language (CPL).
1 Drivers for CPL Based Service Configuration System

1.1 Problem at hand

As voice over IP is becoming popular, end users of this technology are expecting better call services. A wide range of call services (or supplementary services) are possible using VOIP technology like call forwarding, hold, wait etc along with other services such as email, SMS, mobile services and so on. In order to facilitate these end-user requirements, it is a necessity to provision all these call services and allow the end-users to personalize and customize them. This paper addresses this problem. In order to best describe the call service scenarios a standardized approach is necessary so that the service customization can be provided by the service provider rather than the VOIP equipment vendors. A new XML-based language called Call Processing language (CPL) is proposed by IETF for describing and controlling the call scenarios uniquely. This paper proposes system architecture for CPL based application for service configuration.

Fig 1: CPL Based Services on SIP Architecture

1.2 What is CPL?

The Call Processing Language (CPL) is a language that can be used to describe and control Internet telephony services. It is designed to be implementable on either network servers or user agent servers. It is meant to be simple, extensible, easily edited by graphical clients, and independent of operating system or signaling protocol. It is suitable for running on a server where users may not be allowed to execute arbitrary programs, as it has no variables, loops, or ability to run external programs.
The CPL is powerful enough to describe a large number of services and features, but it is limited in power so that it can run safely in Internet telephony servers. The intention is to make it impossible for users to do anything that is more complex (and dangerous) than describing Internet telephony services. The language is not Turing-complete, and provides no means to write loops or recursion.

This document uses SIP terminology, but this should map easily to other systems. Fig 1 shows SIP Architecture that supports CPL based call services. The CPL scripts will be running on the SIP servers that are serving the SIP clients.

1.3 Features of CPL

- **Language characteristics:** These are some abstract attributes of CPL -
  
  - Light-weight, efficient, easy to implement
  - Easily verifiable for correctness
  - Executable in a safe manner - No loops or recursion permitted
  - Easily writeable and parsable by both humans and machines.
  - Extensible - More XML tags can be added as backward compatible enhancements.
  
- **Independent of underlying signaling details:**

  - The same scripts should be usable whether the underlying protocol is SIP, H.323, a traditional telephone network, or any other means of setting up calls.

  - CPL can be extended to processing of other sorts of communication, such as e-mail or fax.

- **Base features -- call signaling:** To be useful, CPL is capable of reaching out to and initiate call signaling events like -

  - Execute actions when a call request arrives

  - Make decisions based on event properties. A number of properties of a call event are relevant for a CPL script's decision process. These include, roughly in order of importance:

    - Destination address
    - Originator address
    - Caller Preferences
    - Information about caller or call
    - Media description
    - Authentication/encryption status

  - Take action based on a call invitation. There are a number of actions that can be taken in response to an incoming call setup request like -

    - Call reject
    - Call redirect
    - Proxy the Call (Forward to another Server)
Call Processing Language (CPL) Based Services Configuration System

- Take action based on a response to a proxied or forked call invitation like -
  - Consider call message fields
  - Relay the call to the call originator
  - For a fork, choose one of several responses to relay back
  - Initiate other actions

- **Base features -- non-signaling:** A number of other features of CPL do not refer to call signaling per se; however, they are still extremely desirable to implement many useful features. The servers which provide these features might reside in other Internet devices, or might be local to the server (or other possibilities). CPL is independent of the location of these servers, at least at a high level.

  - Logging
  - Error reporting
  - Access to user-location info
  - Database access
  - Other external information

- **Language features:** Some features of CPL do not involve any operations external to the CPL’s execution environment, but are still necessary to allow some standard services to be implemented. (This list is not exhaustive.)

  - Pattern-matching: It should be possible to give special treatment to addresses and other text strings based not only on the full string but also on more general or complex sub-patterns of them.

  - Address filtering: Once a set of addresses has been retrieved, the user needs to be able to choose a sub-set of them, based on their address components or other parameters.

  - Randomization: Some forms of call distribution are randomized as to where they actually end up.

  - Date/time information: Users may wish to condition some services (e.g., call forwarding, call distribution) on the current time of day, day of the week, etc. (e.g., working hours, working days based actions)

- **Control:** CPL has a mechanism to send and retrieve other CPL scripts, and associated data, to and from a signaling server. The underlying protocol shall be able to implement these methods and the control of these will be provided to CPL.

### 1.4 Applicability

Broadly, there are two application areas:
- Developing CPL parser residing at VOIP gateways/servers that calls APIs for call events and takes appropriate actions as per the personalized services defined in CPL scripts.

- Developing user interfaces for enabling end users to describe the services and
generate CPL script. Different kinds of user interfaces that can be supported -
- Web based system (For VOIP end-users)
- GUI based system (For call center operators who can customize the services on behalf of end-users)

1.5 Benefits of using CPL

Benefits of CPL based service configuration for the equipment vendors

- No efforts (and cost) required for adding the call service related features from planning phase to deployment phase. Hence cost savings for the vendors.

- Out of the possible call services, few of them will be the more popular services. The vendors need not have to emphasize on this as the personalization of the call services will be more and more end-user dependent.

- Much reduced war on providing different combinations of call services as this is a long list and the competitor may have chosen other set of combinations of services and this particular vendor may not be providing them.

Benefits of CPL based service configuration for the VOIP service providers

- Increase revenues due to customization and personalization of call services.

- Few of the call services will become the more popular out of the possible combinations. The service providers need not have to emphasize on this as the personalization of the call services will be increasingly end-user dependent.

1.6 Open Issues

- CPL is still in draft stage. Till now, 6 drafts have been proposed but none of them is converted to RFC. It means the CPL is still in evolving stage. The changes coming out of these drafts are more or less extensions and enhancements. Architecture framework is in place.

- APIs of SIP and H.323 for application are not standardized. One reason for this is that these APIs are local in nature and are very much vendor specific. But, a general list of the parameters and their data types are known. The open ended APIs can be designed so that the rework on this will not be much. Also, JCC APIs can be used for this purpose.

Fig 2: Interface of CPL based Service Configuration System Components
2 Proposed System Architecture

As discussed in section 1.4, broadly, there are three parts of CPL based Service Configuration application -

- Developing user interfaces for enabling end users to customize the call-based services and generate CPL script for them
- Development of Service Provisioning System that interacts with the service configuration user interface and store it on the server
- Developing Service Invocation System that calls APIs for call events and takes action as per service configured whenever any call event is reported

Architecture for the client-server system is explained in following subsections. Fig 2 shows interface between these three components.

2.1 Service Configuration User interface

End users of VOIP services may not be technical experts who could write the CPL scripts for customizing their services. The Service Configuration User Interface is a user interface for simplifying the CPL script generation. It is a fully secured and authenticated user interface that enables the end user to customize and personalize the call services. This client can be a web based system that can be put on the service provider's web page or it can be a GUI based system running at a call center and is used by operators inside the premises of the service provider.

Fig 3: Architecture of Service Configuration User Interface
The architecture of Service Configuration User Interface is given in fig 3. It consists of following modules:

- Graphical User Interface (GUI)
- CPL Script Generator
- Authentication and security module
- Server Communication module

2.1.1 Graphical User Interface (GUI)

This module performs following functions:

- Provide an interface to the end users or call operators to describe the call services that are expected by the end user.

- It can be a web based interface or a standalone GUI that has various forms and screen for authenticating the GUI user and providing different interactive and simple means to understand so that the expected call services can be customized.

2.1.2 CPL Script Generator

This module performs following functions:

- Provides hooks or functions that can be called on different GUI events (like mouse button click, enter key press etc).

- On occurrence of such events, these functions or hooks will analyze the details provided by the user and generate a CPL script that best describes the call service scenarios entered by the user of GUI.

2.1.3 Authentication and security module

This module performs following functions:

- This module is responsible for collecting authentication information from the end user of the GUI and interacts with the Service Configuration Agent at Service Configuration Provisioning System.

- This will be the first screen / form at GUI at the startup.

- This module will also participate in the security functions (which may require the authentication information) that are executed when every communication happens with the Service Configuration Agent.

2.1.4 Server Communication module

This module performs following functions:

- This module is responsible for providing reliable means for communication between Service Configuration Agent and Service Configuration User Interface.
2.2 Service Invocation System

The main task of this server application is that whenever any signaling message at application level (SIP / H.323) is received for a call then identifies the subscriber for which the call is, if required, identify CPL script(s) for that user and execute them as per that call actions.

The Service Invocation System is divided into following modules viz:

- Call-Specific event generator
- CPL Parser
- CPL script analyzer
- Service APIs with Database interface

Refer fig 4 for architecture of JCC based Service Invocation System among these modules.

If the switch is implementing the personalization of VOIP service only (that is the switch does not support protocols like ISUP, PRI, INAP etc.) then a light-weight cost effective solution could be based on SIP Servlets. The SIP Servlets provides standardized access to SIP stack and allows the applications to register for specific events, initiates the requests or respond to the registered events. There can be multiple applications that can receive the same event. Refer fig 5 for architecture of SIP Servlet based Service Invocation System.
2.2.1 Call-Specific event generator

This module performs following functions:

- Accesses signaling protocol APIs to determine all kinds of call events that are required for the CPL scripts (like JCC APIs or SIP Servlets). There may be other sources for event generation like timer expiry.

- Ascertain targeted end-user information from the event in order to find out CPL script(s).

- Find CPL script(s), if any, from the database and submit them to CPL Parser module.

2.2.2 CPL Parser

This module performs following functions:

- Parse the CPL script
- Call appropriate backend functions from CPL script analyzer for handling the call events
2.2.3 CPL script Analyzer

This module performs following functions:

- Analyze the CPL script
- Take actions as required by calling appropriate Signaling Server APIs or Database APIs
- Start timer(s) if required by the CPL script (for events like "after 4 rings")
- Report errors, if any

2.2.4 Service APIs with Database interface

This module performs following functions:

- Provide APIs for database interface
- Enables the actions to be taken using service APIs of underlying signaling protocols using JAIN Service Logic Execution Environment (JSLEE) with JAIN Call Control (JCC)

This module is actually part of the VOIP gateway and signaling protocols running on it. It can be implemented in the form of JCC APIs. These APIs must have been existing on the underlying VOIP servers. If not, then they need to be developed.

2.3 Service Configuration Provisioning System

Service Configuration Provisioning System will reside on VOIP Gateway or Proxy servers. This system is responsible for storing user specific CPL scripts for describing their services that are generated by the service configuration user interface.

The Service Configuration Provisioning System is divided into following modules viz:

- Service Configuration Agent
- Signaling Server APIs and Database interface

Refer fig 6 for architecture of CPL parser that gives interfaces among these modules.

![Architecture of Service Configuration Provisioning System](image)
2.3.1 Service Configuration Agent

This module is a typical server module in client/server architecture which is responsible for communicating with the service configuration user interface and call appropriate APIs of Signaling Server APIs and Database interface.

2.3.2 Service APIs and Database interface

Refer section 2.2.4 for details about this module.

3 Alternate Technologies Available in This Space

3.1 Other Technologies

Other alternative solutions for the same problem domain are:

- SIP CGI: SIP CGI [4] is an interface for implementing services on SIP servers. Unlike a CPL, it is a very low-level interface, and would not be appropriate for services written by non-trusted users. Also, SIP CGI can be implemented in languages (like C, c++) that allow recursion, complex loops. Hence, there is no guaranty that the CGI will terminate in real time. This aspect is very important as it can affect call setup time. Refer table 1 for detailed comparison of CPL with SIP CGI.

- Scripting Languages such as Perl, Tcl: All the existing scripting languages are, naturally, expressively complete; this has two inherent disadvantages. Any function implemented in them can take an arbitrarily long time, use an arbitrarily large amount of memory, and may never terminate. For call processing, this sort of resource usage is probably not necessary, may be undesirable.

- Java Applet: The difficulty with Java Applets is primarily its lack of transparency and portability. Also, unless the levels of these bounds are imposed by the standard, a bad idea so long as available resources are increasing exponentially with Moore's Law, a user can never be sure whether a particular program can successfully be executed on a given server without running into the server's resource limits, and a program which executes successfully on one server may fail unexpectedly on another.

- Vendor Specific proprietary solutions
### 3.2 Pros and Cons of using CPL

#### Advantages of CPL

- CPL is easily parsable, unlike the existing scripting languages that make automatic generation and parsing of the scripts very difficult, as every parsing tool must be a full interpreter for the language.

- CPL has advantages over another alternative, Java Applets. The difficulty with Java Applets is primarily its lack of transparency and portability.

- CPL syntax is like XML. The text markup languages like XML can be easily manipulated by smart editors, powerful document programming languages such as LaTeX or Postscript usually cannot be. This makes CPL an easily parsable language, requiring very less resources (time and memory) for parsing.

- XML parsers are freely available in C, C++, and Java. They can be used for building CPL parsers.

- Remotely accessible web based interfaces or GUI are possible for auto-generation of scripts for capturing user-friendly and customized services.

- Standardized approach for defined customized services.

- CPL is OS and signaling protocol independent.

#### Disadvantages of CPL

- CPL is still in evolving phase.
• Care has to be taken while writing (or auto-generating using user interface) the CPL scripts so that any conflicting call behaviors are not present.

• Current CPL draft [5] is limited to support tags which are generic in nature, but support events that are very much similar to SIP and H.323. Other technology support is not discussed in CPL draft. But the CPL scripts can be extended.

4 Applications

4.1 Example Call Services

This section gives some specific examples of services which the end-users will be able to create by using CPL scripts.

Example 1: Call forward on busy/no answer
When a new call comes in, the call should ring at the user's desk telephone. If it is busy, the call can always be redirected to the user's voicemail box. Suppose there's no answer after four rings, it can also be redirected to his or her voicemail, unless it's from a supervisor, in which case it should be proxied to the user's cell phone if it is currently registered.

Example 2: Information address
A company advertises a general "information" address for prospective customers. When a call comes in to this address, during working hours, the caller will be given a list of the people who are willing to accept general information calls. If it's outside of working hours, the caller can get a web-page indicating what times they can call.

Example 3: Intelligent user location
When a call comes in, the list of locations where the user has registered should be consulted. Depending on the type of call (work, personal, etc.), the call should ring at an appropriate subset of the registered locations, depending on information in the registrations. If the user picks up from more than one station, the pick-ups should be reported back separately to the calling party.

Example 4: Intelligent user location with media knowledge
When a call comes in, the call can be proxied to the station that the user has registered from whose media capabilities best match those specified in the call request. If the user does not pick up from that station within four rings, the call should be proxied to the other stations from which he or she has registered, sequentially, in order of decreasing closeness of match.

![Fig 7: Example Call Service Scenario](image-url)
Fig 8: CPL Script for Example Call Service Scenario

Example 5: Client billing allocation - lawyer’s office
When a call comes in, the calling address is correlated with the corresponding client, and client’s name, address, and the time of the call is logged. If no corresponding client is found, the call is forwarded to the lawyer’s secretary.

This list of examples is not complete. A customized service scenario based on end-user needs can be described unambiguously using CPL. Refer fig 7 for a sample case and the corresponding CPL script for this scenario is given in fig 8.

4.2 Usage Scenarios

A CPL would be useful for implementing services in a number of different scenarios. A few examples are given below:

- **Script creation by end user:** In the most direct approach for creating a service with a CPL, an end user simply creates a script describing the expected service. He or she simply decides what service he or she wants, describes it by using a CPL script, and then uploads it to a server.
- **Third party outsourcing**: Because a CPL is a standardized language, it can also be used to allow third parties to create or customize services for clients. These scripts can then be run on servers owned by the end user or the user's service provider.

- **Administrator service definition**: A CPL can also be used by server administrators to create simple services or describe policy for servers they control. If a server is implementing CPL services in any case, extending the service architecture to allow administrators as well as users to create scripts is a simple extension.

- **Web middleware**: Finally, there have been a number of proposals for service creation or customization using web interfaces. A CPL could be used as the back end to such environments: a web application could create a CPL script on behalf of a user, and the telephony server could then implement the services without either component having to be aware of the specifics of the other.

**Conclusion**

As call services are getting popular, the end-users are expecting more number of services and personalize them as per their needs. This paper proposed a solution for this, which uses CPL as a basic mean to support personalization of the services by the end-users. The basic architecture of the system is comprised of - Service configuration client and Service configuration server. The Service configuration client is a graphical user interface which can also be made web based that allows the end users to configure the services. The Service configuration server consists of agent interface for the client that allows storing of the call services for each end user in the form of CPL scripts. It also contains call event generator and CPL parser that calls the basic call service APIs. This architecture uses the advantages of CPL, client-server architecture that addresses issues like security and authentication. The architecture is also light weight so that real-time processing is faster. The proposed architecture is based on Standard interfaces for call control and execution environment like SIP Servlet, JCC and JSLEE.

**Acronyms and References**

**Acronyms**

- CGI: Common Gateway Interface
- CPL: Call Processing Language
- DTD: Data Type Definition
- GUI: Graphical User Interface
- IETF: Internet Engineering Task Force
- IN: Intelligent Network
- INAP: Intelligent Network Application Part
- IPTEL: IP Telephony
- ISUP: ISDN User Part
ITU International telecommunication Union
JAIN JAVA APIs for Integrated Networks
JCC JAIN Call Control
MAP Mobile Application Part
MGCP Media Gateway Control Protocol
SIP Session Initiation Protocol
TCAP Transaction Capability Application Part
XML Extensible Markup Language
VOIP Voice Over IP

References


About the Authors

Mahavir D Karnavat is specialist in telecom with special interest in VOIP and 3G Wireless protocols and has been involved in design and development of SIP applications and protocol stacks. Currently he is Technical Lead in Wipro’s Telecom and Inter-networking division.

Shivaji Hogale is Project Manager with experience in various telecom technologies including legacy products and 3G wireless protocols.
About Wipro Technologies

Wipro is the first PCMM Level 5 and SEI CMMi Level 5 certified IT Services Company globally. Wipro provides comprehensive IT solutions and services (including systems integration, IS outsourcing, package implementation, software application development and maintenance) and Research & Development services (hardware and software design, development and implementation) to corporations globally.

Wipro's unique value proposition is further delivered through our pioneering Offshore Outsourcing Model and stringent Quality Processes of SEI and Six Sigma.

Wipro in T&I

Wipro T&I offers comprehensive solutions for telecommunication to confront challenges, and convert every challenge into an opportunity. With over two decades of Telecom experience, Wipro T&I offers a wide range of solutions in the following domains:

- Wireless Networking
- Broadband ((Data, optical and access networking)
- Voice Switching
- Network Management
- Hardware Design

Wipro T&I also provides several IPs, Components and Reference Solutions in these domains which help our customers in reducing Time to Market and Save Costs. We offer complete Consulting, Architecting, Design, Implementation and Maintenance Services to the Telecom Equipment Manufacturers. We do not sell or manufacture any products, as we are very focused in the R&D and Engineering Design Services Outsourcing.

© Copyright 2002. Wipro Technologies. All rights reserved. No part of this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without express written permission from Wipro Technologies. Specifications subject to change without notice. All other trademarks mentioned herein are the property of their respective owners. Specifications subject to change without notice.

America
1995 El Camino Real, Suite 200
Santa Clara, CA 95050, USA
Phone:+1 (408) 2496345
Fax: +1 (408) 6157174/6157178

Europe
137, Euston Road
London NW12AA,UK
Phone:+ (44) 020 73870606
Fax: + (44) 020 73870605

Japan
Saint Paul Bldg, 5-14-11
Higashi-Oi, Shinagawa-Ku,
Tokyo 140-0011Japan
Phone:+(81) 354627921
Fax: +(81) 354627922

India-Worldwide HD
Doddakannelli, Sarjapur Road
Bangalore-560 035, India
Phone:+ (91) 80 8440011 -15
Fax: +(91) 80 8440254

www.wipro.com
eMail: info@wipro.com

For more white papers log onto www.wipro.com/shortcuts/downloads.htm