Simulation of Personalized Services in SIP Communications

Dongmei Jiang[1], Tet Hin Yeap[1], Luigi Logrippo[1,2]

[1] SITE, University of Ottawa, 800 King Edward Ave., Box 97, Ottawa, Ontario, K1N 6N5, Canada
[2] Université du Québec en Outaouais, Département d'informatique et ingénierie
[1] {djiang, tet, luigi@site.uottawa.ca}

***Abstract*- A web-based system is designed and implemented to simulate advanced services in a Session Initiation Protocol (SIP) environment, including system basic services and personalized services (policies). Described in extended Call Processing Language (CPL), personal policies are programmed by end users via Graphic User Interfaces (GUIs) and are automatically translated into extended CPL. The simulation system presented in this paper clearly displays what CPL policies should be used for the provision of presence services and call-processing services. Policy conflicts can be addressed by setting policy priority in the simulation system.**

***Keywords:* SIP; presence; extended CPL; personalized services (policies); service conflicts**

I. INTRODUCTION

As a signalling protocol, the Session Initiation Protocol (SIP) [1][2] can establish, modify and terminate multimedia sessions or calls. Built on top of the transport layer, SIP can support many new services including multimedia communications and presence. Presence [3] conveys the willingness and the ability of a user to communicate with others on a network. With the creation of various new services, service management (creation, description, control and provision) becomes more complicated. The Call Processing Language (CPL) [4] was proposed by the Internet Engineering Task Force (IETF) to describe personalized telephony services and was extended to include presence related services [5]. Through a simple and clear service management system described in this paper, a simulation system is designed to experiment with service descriptions in extended CPL and to demonstrate SIP communication services (especially personalized services for call-processing and presence). This system provides a simulation environment that can be used as a basis for further research. Service interactions (conflicts) are addressed as well.

II. SERVICE AND SERVICE DESCRIPTION

Services in the simulation system can be of two types: system basic services and personalized services. System basic services are provided to all users equally. For example in a call-processing system (or a presence system), basic services include sending requests for a caller (or a watcher in a presence system) or replying to the request for a callee (or authenticating and authorizing the watcher's request in a presence system, and notifying the watcher once the request is approved). Personalized services (policies) are provided to satisfy user's specific needs. They are associated with and owned by a particular user and will be triggered only when the request is for the user. For example, Alice will be able to prevent her boss from knowing her presence status outside work hours.

CPL can work on top of either SIP from the IETF or H.323 [6] from the International Telecommunications Union-Telecommunications Standard Sector (ITU-T). This paper only focuses on the incorporation of CPL with SIP. CPL can process SIP INVITE [1] messages and deal with call-processing only. After the extension of presence information [5] from the well know "online" and "offline" indicators to include "location", "LineStatus", "role" and "availability", CPL was further extended [5] to process SIP SUBSCRIBE and SIP NOTIFY [2] messages for presence related services. With these extensions, services can be handled with consideration of a person's presence status, time, address or any of their combinations in both call-processing systems and presence systems.

```
<cpl xmlns = ?   厩
<cplPresence:incoming-notification>

 <address-switch field="origin">
 <address is="sip:Sharon@example.com">

  <cplPresence:presence-switch presentity ="sip:Sharon@example.com">
  <cplPresence:presence location = "office">
   <cplPresence:success>
    <location url="sip:Sharon@example.com">
      <cplPresence:call/>
    </location>
   </cplPresence:success>
  </cplPresence:presence>
  </cplPresence:presence-switch>

 </address>
 </address-switch>

</cplPresence:incoming-notification>
<cpl>
```

Fig.1. Presence-based Auto Calls

Fig. 1 shows an example of Peter's policies in extended CPL. Peter requests to initiate an auto call to Sharon as soon as he is notified that Sharon arrives at her office. This auto call is based on the callee's (Sharon's) presence status that is carried in the SIP NOTIFY from Sharon to Peter. Such personalized services with rich presence are realized in our simulation system. In this example, Peter plays the role of "watcher" and Sharon plays the role of "presentity" in the presence system, while Peter plays the role of "caller" and

Sharon plays the role of "callee" in the call-processing system.

III. SYSTEM DESIGN AND IMPLEMENTATION

The web-based simulation system for presence services is designed and implemented by using the Java programming language. Focusing on presence and call-processing simulation, the system shown in Fig. 2 contains three subsystems: the presence system, the call-processing system and the policy system. The name of each subsystem is a link leading users to the corresponding subsystem.

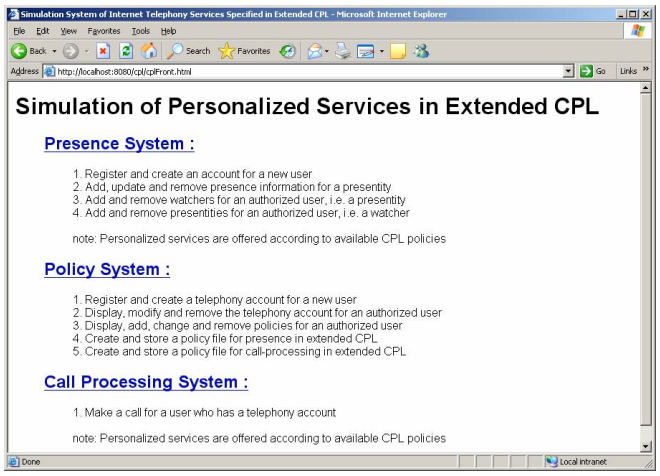

Fig. 2. Simulation System

The three subsystems can either be independent of each other or cooperate with each other. The call-processing system allows a registered user to make phone calls to other registered users. The presence system can deliver presence information to watchers; register, modify and remove presence information for a presentity; add and remove watchers for a presentity; add and remove presentities for a watcher. The policy system can create and modify user telephone accounts; create, modify and remove user policies; translate user policies into extended CPL and store the CPL files in the system. Personalized services in either a presence system or a call-processing system will be offered according to the CPL policies respectively. System basic services will be provided only if personal CPL policies are not available.

A user needs to create an account in order to login to these subsystems. A user is identified by a unique name with a confirmed password. The user is allowed to have a logical phone, a regular phone, a cell phone and a voice mail, each of which is identified by an address consistent with SIP. These addresses are valuable elements that will be extracted and written into the user's CPL policies.

The system architecture, shown in Fig. 3, contains four parts: the Internet browsers, the web server, the database server and the database. The Internet browsers let users enter their requests and display request results. The web server holds Java servlets that are the central controllers managing the simulation services. The Database server holds the database agent who works as a representative for the database and has various methods to operate on the database. The database contains all service-required information. User requests are entered via the browser's Graphic User Interfaces (GUIs). The Java servlets accept the request through HTTP and initiate a corresponding request to the database agent that is connected to the database via the Java Database Connection (JDBC). The database agent obtains the required data from the database and sends it back to the Java servlets. The Java servlets process the data and send the readable service results to the user browser. Held by their own servers at different locations as shown in Fig. 3, the database agent and the Java servlets are independent of each other in this architecture, which makes them easy to extend and modify individually.

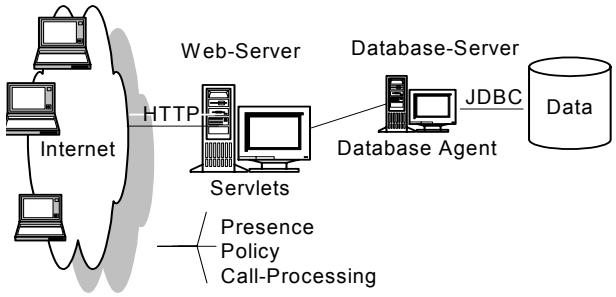

Fig. 3. Simulation System Architecture

IV. POLICY MANAGEMENT

The policies are classified into six types to process three kinds of SIP messages (INVITE, SUBSCRIBE and NOTIFY) with "incoming" and "outgoing" directions considered. They are incoming call (IN) and outgoing call (OUT) in a call-processing system; and incoming subscription request (SIN), outgoing subscription request (SOUT), incoming notification response (NIN) and outgoing notification response (NOUT) in a presence system.

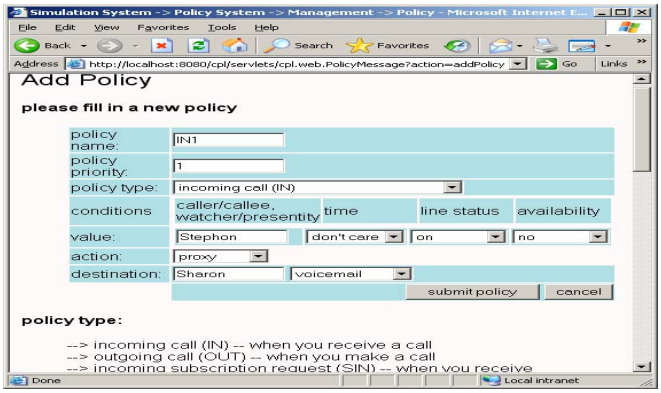

Fig. 4. Policy Creation

A CPL policy in the simulation is composed of the following four parts: type (CPL top action), conditions (CPL switches), one action (CPL action) and destination (CPL location). This can be clearly reflected on the policy creation GUI, shown in Fig. 4. Through the completely designed GUI,

users can specify service type and their personalized needs in terms of a person's status, time, address or any of their combinations.

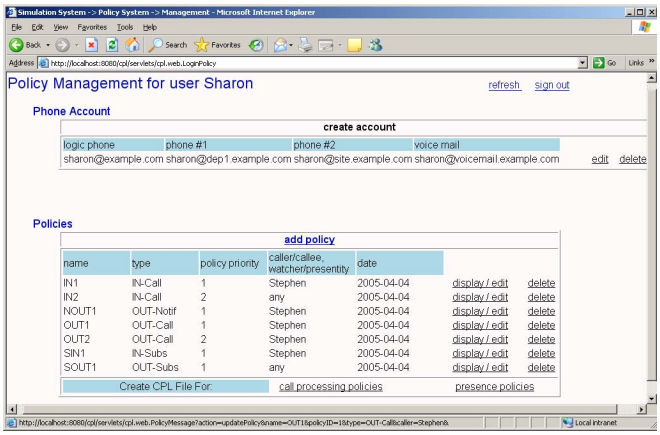

Fig. 5. Policy Management

```
<cpl ..... >
<cplPresence: incoming subscription>
  <!-- ----  Policy # 1 - SIN1  ------ -->
  <address-switch field = "orgin">
  <address is = "sip:stephen@example.com">
    <cplPresence:presence-switch presentity="sip:sharon@example.com">
    <cplPresence:presence lineStatus = "on" >
    <cplPresence:success>
      <reject/>
    </cplPresence:success>
    </cplPresence:presence>
    </cplPresence:presence-switch >
  </address >
  </address-switch >
</cplPresence:incoming subscription>

<cplPresence:outgoing subscription>
  <!-- ----  Policy # 1 -- SOUT1  ------ -->
    <time-switch tzid="America/New-York"
              tzurl="http://example.com/tz/America/New_york">
    <time dtstart="20030101T180000" duration = "PT14H"  freq = "weekly"
        byday = "MO, TU, WE, TH, FR">
      <reject/>
    </time>
    </time-switch >
</cplPresence:outgoing subscription>

<cplPresence:outgoing notification>
  <!-- ----  Policy # 1 -- NOUT1  ------ -->
  <address-switch field = "orgin">
  <address is = "sip:stephen@example.com">
     <cplPresence:presence-switch presentity="sip:sharon@example.com">
     <cplPresence:presence lineStatus = "on" >
     <cplPresence:success>
       <reject/>
     </cplPresence:success>
     </cplPresence:presence>
     </cplPresence:presence-switch >
  </address >
  </address-switch >
</presence: outgoing notification>
</cpl>
```

Fig. 6. Auto-created Presence Policies in extended CPL

The policy management GUI will pop up after a user submits his policy. Fig. 5 is the management GUI for user Sharon. By clicking the account related buttons "edit" and "delete", Sharon can edit her personal account and deregister her personal information from the system respectively. Using policy related buttons "add", "edit", "display" and "delete", Sharon can add, edit, display and remove her policies. Using the buttons for policy creation ("call processing policies" and "presence policies"), Sharon's policies are automatically translated into extended CPL files and stored in the database of the simulation system. These CPL policies will be executable for both Sharon's call-processing services and her presence services.

The CPL files for the presence policies and the call-processing policies are separately created and stored and they will be executed individually in two different subsystems. Fig. 6 is the CPL file for Sharon's presence policies. For example, policy "NOUT1" indicates that Sharon blocks notifications to Stephen if she is on her phone.

One user can have multiple policies for each policy type. In Fig. 5, Sharon has two incoming call policies and two outgoing call policies. Sometimes there exist policy conflicts. For example, Bob prefers to forward his incoming calls to his voice mail from 9:00am to 10:00am, however, he wants to take his wife's calls unconditionally. A conflict occurs if Bob's wife calls him at 9:30am. Giving a higher priority to the policy for his wife's calls, the conflict is solved and Bob is able to take his wife's calls even at 9:30am In order to eliminate policy conflicts, policies are arranged by numbers when created as shown in Fig. 3: the smaller the number, the higher the policy priority. The policy priority order inside each service type is clearly displayed in the policy management GUI as shown in Fig. 5. In the processing of the policies for each service, the highest priority policy is checked first. As soon as a policy matches the criteria, this policy is executed and the remaining policies are ignored.

The study of such service conflicts (sometime called feature interactions in the telephony world) is beyond the scope of this paper. However clearly they will have to be taken into careful consideration by the system designers. In many cases, the naive user unfortunately will not be able to distinguish interactions from system malfunctions. Reviews of research on service conflicts in traditional telecommunications systems can be found in [7][8].

V. PRESENCE MANAGEMENT

In the presence system, a user (e.g. Sharon shown in Fig. 7) can be a watcher, a presentity or both at the same time. As a watcher, Sharon can manage her presentities; as a presentity, Sharon can manage her watchers and notify them of her presence information.

The presence management GUI, shown in Fig. 7, will be popped up when the user (Sharon) logins to the presence system. A user's presence status is characterized by the parameters "location", "lineStatus", "role" and "availability". The presence information, stored in the system database, can be checked up while executing the related CPL policies.

Each button in Fig. 7 connects to a presence service on the GUI. For example, by clicking the button "add watcher", the GUI "Add Watcher" is popped up allowing the current user Sharon to approve an incoming subscription request from a

specified watcher. By clicking the button "add presentity", the "Add Presentity" GUI is popped up allowing Sharon to send out a subscription request to the specified presentity. Sharon can change her presence status via the GUI "Presence Update", which is popped up by clicking the button "edit". She may deregister her presence information from the system by clicking the button "delete". Sharon is allowed to terminate her presence service to her watcher (e.g. Dongmei) by clicking the button "delete" in the row of watcher Dongmei. Sharon also can stop being informed of the status of her presentity (e.g. Christopher) by clicking the responding button "delete".

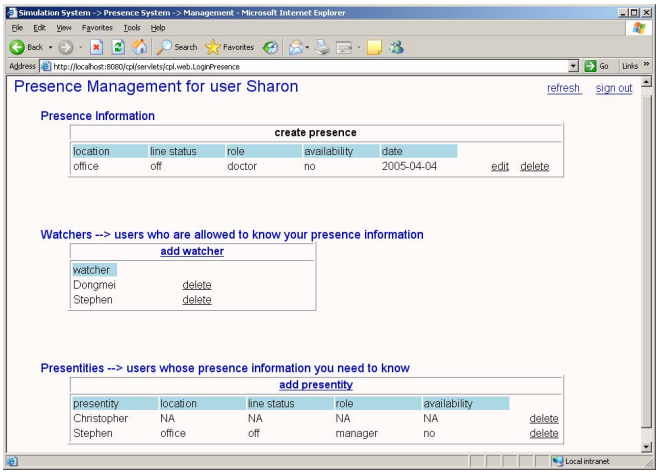

Fig. 7. Presence Management

The presence system will provide user specific services (policies) once a user has his personal CPL policies. In the example of Sharon, Sharon has a conditional outgoing notification policy "NOUT1", shown in Fig. 6, to block her notifications to Stephen when she is talking on her phone during working hours.

A CPL policy is triggered by a service event in the simulation. When Sharon updates her phone line status from "off" to "on" in working hours via the GUI "Presence Update", the result for her presence update event is shown in Fig. 8. Stephen is not notified according to Sharon's policy "NOUT1". The other watcher (Dongmei), for whom Sharon has no notification policy, is successfully notified in the system default behavior.

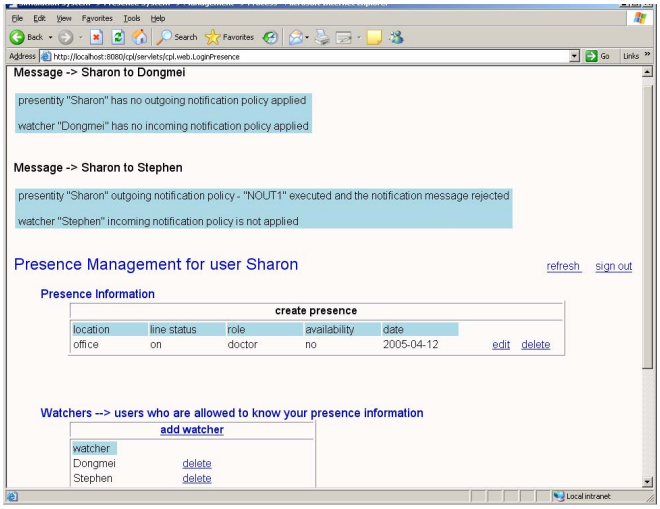

Fig. 8 Policy "NOUT1" Provision

The above test case clearly illustrates what CPL policies should be used for the provision of presence services. In the call-processing system, the personal CPL policies are provided very similarly to the presence system. If no personal policies are available, system basic services are provided automatically.

VI. CONCLUSION

This paper describes a system to simulate SIP communications with personalized services expressed in extended CPL. Via the GUIs, end users can program their specific services, which are translated into extended CPL automatically by the system. Presence services and call-processing services are then simulated according to user's policies if available. Policy conflicts are solved by giving a priority to each policy. This simulation environment can be used as a basis for further research.

REFERENCES

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, etc. "SIP: Session Initiation protocol", IETF RFC 3261, June 2002.

[2] A.B. Roach, "Session Initiation Protocol (SIP)-Specific Event Notification", IETF RFC3265, June 2002.

[3] J.Rosenberg, "A Presence Event Package for the Session Initiation Protocol (SIP)", IETF RFC 3856, August 2004.

[4] J. Lennox, X. Wu, H. Schulzrinne, "Call Processing Language (CPL): A language for User Control of Internet Telephony Services", IETF RFC3880, October 2004.

[5] D. Jiang, Internet Telephony Services for Presence with SIP and Extended CPL, Master's Thesis, University of Ottawa, 2004.

[6] Ismail Dalgic and Hanlin Fang, "Comparison of H.323 and SIP for IP Telephony Signalling", Proc. Of Photonics East, Boston, Massachusetts, Sep. 1999.

[7] M. Calder, E. Magill, M. Kolberg, S.Reiff-Marganiec. "Feature Interaction: A Critical Review and Considered Forecast". Computer Networks, Volume 41/1, 2003.

[8] J. Cameron and H. Velthuijsen, "Feature Interactions in Telecommunications Systems". IEEE Communications Magazine, Vol. 31, no. 8, 18-23, August 1993.