SIPC, a multi-function SIP user agent

Xiaotao Wu, Henning Schulzrinne

Columbia University, Department of Computer Science,
New York, New York 10032, U.S.A.
{xiaotaow, hgs}@cs.columbia.edu

Abstract. Integrating multiple functions into one communication user agent can introduce many innovative communication services. For example, with networked appliance control, a user agent can turn off the stereo when receiving an incoming call. With location sensing, a user agent can automatically reject a call if it knows the location preference is 'quiet'. Multi-function interactions enable services that are otherwise impossible. In this paper, we first present the new services introduced by the integration, then introduce our SIP user agent, SIPC, which handles these new services in a programmable way. SIPC integrates multimedia call setup, networked appliance control, presence handling, Internet TV, instant messaging, location sensing, networked resource discovery, third-party call control, real-time multimedia streaming, emergency call handling, and conference floor control into one application. We analyze the relationship among these functions and propose different approaches for function integration. SIPC uses the Session Initiation Protocol (SIP) for multimedia call setup and the Language for End System Services (LESS) for service programming.
Keywords: multi-function integration; SIP; SAP; networked appliance control; location-based services; SLP; RTSP; SIP event notification; floor control; LESS

1 Introduction

One of the most important advantages of Internet telephony is its ability to provide innovative services. In Internet telephony systems, traditional telephony services, such as call transfer, can be enhanced by the integration of Internet services, such as email, web, instant message, presence notification and directory lookups. The enhancements require Internet telephony end systems to perform more functions in addition to audio and video communications.

Some instant messaging applications, such as MSN Messenger, have integrated online/offline indication, instant messaging, email, and web browsing into one application. In our SIP [1] user agent, SIPC [2], in addition to the functions mentioned above, we also support networked appliance control, real-time multimedia streaming, networked resource discovery, third-party call control, Internet TV, location sensing, emergency call handling, and conference floor control. Multiple functions may interact with each other and introduce many new services that are otherwise impossible. For example, SIPC can automatically turn off the stereo in the user's room when receiving an incoming call. A SIPC user can share an Session Announcement Protocol (SAP) session with his friends by putting the session information in a SIP INVITE request. SIPC can base call decisions on location information. In Section 2, we detail the new service examples.

Too many functions in one application may make the application too complicated to maintain. In Section 3, we analyze the relationship among all the functions in SIPC. Based on the analysis, we discuss the practical integration approaches that can minimize the overall application complexity, while still providing convenient ways for function interaction.

SIPC handles multi-function interaction in a programmable way. We have defined a service creation scripting language called the Language for End System Services (LESS) [3]. In Section 4, we describe how SIPC uses LESS scripts to perform multi-function interaction.

In Section 5, we briefly introduce the implementation details of SIPC. Section 6 concludes the paper and discusses our future work.

2 New services introduced by multi-function integration

In traditional telephony systems, communication services are provided by the switches in communication networks. The services are performed based on very limited information, such as the address and the busy status of the caller and the callee, allowing only a small set of actions, in most cases, to route calls.

In Internet telephony systems, services can be implemented in both network servers and intelligent end systems. With the integration of Internet services, such as presence indication, Internet telephony services have access to much richer information, and offer a richer set of service actions, not limited to call routing, the actions can also be networked appliance control, instant messaging, email and web browsing. We describe a few of the new services below.

2.1 Setup preferable communication environment

Communication quality is not only determined by the quality of audio/video streams transmitted between endpoints, but also affected by the communication environment where the talkers are in. For example, background noise may affect audio conversation and brightness of lights may affect video conversation. In a networked home with network controllable appliances, the integration of networked appliance control into a communication agent may help to setup environment conducive to communication. In our lab environment, SIPC can automatically pause the stereo through a Slink-e controller [4] when receiving an incoming call. If the call requires video communication, SIPC can also automatically adjust the brightness of the lamp in our lab through an X10 controller. SIPC uses the SIP DO [5] method to perform networked appliance control.

2.2 Call handling based on presence information

The integration of presence information handling can help to make call decisions. In traditional telephony systems, a caller usually knows nothing about a callee's status before making a call. In Internet telephony, a caller can know not only the online/offline status, but also other information, such as the location privacy preference, of the callee. SIPC can generate many new services based on the status information, for example,

automatically calling a friend when the friend is online, or starting a conference only when all the essential participants are online.

2.3 Use networked resources

The integration of location sensing, networked resource discovery, networked appliance control, and third-party call control enables a portable end system to use networked resources for better communication quality. Usually the capability of an end system is in inverse proportion to its portability. A portable end system usually has a small display, low-quality audio, and inconvenient input devices. However, if there are networked devices with good multimedia I/O capabilities in the communication environment, user agents with the support of Service Location Protocol (SLP) [6] and SIP third-party call control architecture (3pcc) [7] can control the networked devices for communication.

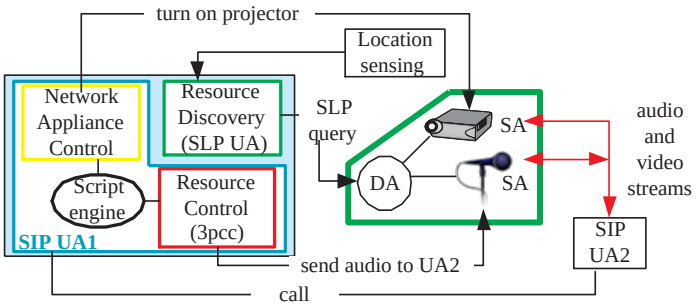

Fig. 1. Using networked resources for better communication quality

We have proposed an architecture [8] [9] that allows end systems to use available resources in the environment, such as displaying video on a wall-hanging plasma display or getting audio from an echo-canceling microphone. To support such an architecture, an end system needs to find out available resources and control them. As shown in Figure 1, with location sensing, SIPC may retrieve location information and find available resources in the communication environment by including location information in the SLP query [6]. SIPC can then use networked appliance control and SIP third-party call control (3pcc) [7] to control the resources.

2.4 Location sensing and location-based services

Many applications used in the Internet today benefit from using location information. In Internet telephony systems, location information may help to make call decisions or trigger automatic communication actions. For example, when receiving an incoming call, SIPC can be programmed to check its own location and then play a loud ring tone

if the place-type is street, or flash its icon if the privacy of the place is quiet. SIPC uses location information in three ways: it becomes part of outgoing requests sent to remote parties, it triggers automatic actions, and it governs communication behaviors.

Location information can be revealed to remote parties for location tracking as part of the presence notification or encoded in MIME [10] with other content. Location information could be room (name or function information), civic (street and community), categorical (such as movie theater), activity (such as travel) and privacy preference (such as quiet). SIPC can convey location information, for example, in a SIP NOTIFY request in RPID format [11] or GEOPRIV Location Object Format [12], to the parties explicitly showing interests in the information. SIPC can also include the location information in SIP REGISTER or PUBLISH [13] requests to upload the location information to a location server. When sending an emergency call, SIPC will encode its location information in MIME in a SIP INVITE request. The emergency call taker can conveniently track the caller with the location information.

When SIPC gets location information, it may invoke a service script, such as a LESS [3] script, to perform automatic actions. The location information can be the user's own location or remote buddy's location. The service script can handle absolute location information or relative location information between two people. For example, when SIPC gets its own room number, it can automatically turn on the light of the room. When it gets its buddy's location and find the distance to the buddy is less than a certain value, it can automatically send an instant message to the buddy.

Integrating location information with call control services can help to govern appropriate communication behavior. For example, in a movie theatre with a movie playing, the Bluetooth device in the movie theatre may broadcast its location information as 'quiet', when SIPC gets the location information, its service scripts may automatically block incoming calls unless the priority of the call is `emergency`.

2.5 Internet TV session sharing

The Session Announcement Protocol (SAP) [14] advertises multicast multimedia sessions and their parameters to prospective participants. Integrating a SAP user agent into SIPC allows users to easily share an interesting program with their friends. If a user finds an interesting program and wants to ask his friends to watch the same program, the user needs to convey the program information to his friends. Since both SIP and SAP use the Session Description Protocol (SDP) [15] to describe session information, SIPC can get the SDP content of the SAP packets and put the content in a SIP INVITE request. This way, the user can simply call his friends with the SDP content without having to know the session details.

2.6 Voicemail handling

The integration of web, email and SIP message waiting indication [16] provides various ways for handling voicemail. A voicemail can be sent as an email attachment, or as a HTTP [17] URL or a Real Time Streaming Protocol (RTSP) [18] URL in an email.

The voicemail information can also be in SIP message waiting indication[16] notification. SIPC can dial into the voicemail server to get the voicemail, or play the email attachment, or start a web browser to retrieve the voicemail.

2.7 Conference floor control with active talker indicator

During a conference, floor control [19] helps to assign talking rights. Only the floor holders' voice gets delivered to each participant. In a classroom environment, when a student gets the floor, turning on the light on the student's desk, or adjusting the video camera to face the student may help to find the talker. The integration of networked appliance control with conference floor control in SIPC can handle this task gracefully.

3 How to integrate multiple functions

The above service examples show that multi-function integration may bring many innovative services. However, integrating too many functions in one application may make the application too complicated and may confuse users if the application contains functions users don't need. Since SIPC directly interacts with users, any confusion from users may impair its usability. It is very important to choose an appropriate integration method to enable the new services in SIPC but without making it too complicated and without adding too much implementation efforts. Before discussing the integration methods, we first list the functions integrated in SIPC, and investigate the relationship among these functions.

3.1 Functions integrated in SIPC

SIPC can support a range of media types, such as audio, video, whiteboard and desktop sharing and can perform functions beyond multimedia calls. SIPC uses the SIP DO [5] method to perform networked appliance control, uses the SIP event notification architecture [20] to perform presence notification, uses the Session Announcement Protocol (SAP) [14] to retrieve multicast multimedia session information, uses RTSP to retrieve voicemail, uses DHCP Options for Civic Addresses [21] and GEOPRIV Location Object Format [12] for location sensing, uses the Service Location Protocol (SLP) [6] to find available networked resources, uses SIP for third-party call control [7] to control networked resources. SIPC uses external applications to handle email and web browsing. SIPC integrates a SIP CGI [22] and a LESS [3]/CPL [23] engine to handle service script. Section 5 provides more details on the implementation.

When integrating all these functions into SIPC, we noticed that many functions overlap each other, and the functions may interact with each other in different ways. Below we analyze the relationship among these functions and propose the integration approaches based on this analysis.

3.2 Overlap among SIPC functions

Many of the functions mentioned in Section 2 overlap with each other. Because of the overlap, integrating a new function into SIPC will not increase the overall complexity

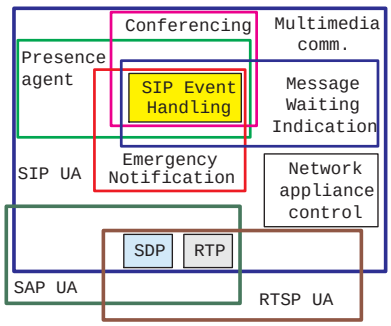

Fig. 2. Overlap among SIPC functions

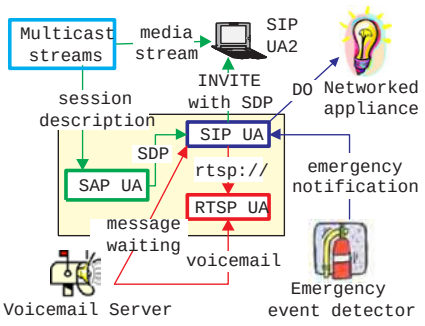

Fig. 3. Interaction among SIPC functions

too much. As shown in Figure 2, SAP user agents, RTSP user agents and SIP [1] user agents all use SDP [15] for session description and RTP [24] for real time media stream transmission. The presence status, conference status, location information, and emergency event can all be transmitted by the SIP event notification architecture [20]. All of the SIP event notification, SIP multimedia session setup and SIP networked appliance control can share the same SIP stack.

3.3 Interaction among SIPC functions

In the service examples described in Section 2, we noticed that multi-function interaction introduces new services. As shown in Figure 3, the SAP user agent passes the session description information to the SIP user agent so the SIP user agent can invite another SIP user agent, SIP UA2, to watch the same multicast media session. When the SIP user agent gets the message waiting indication from voicemail server, it can instruct the RTSP user agent to retrieve the voicemail. When the SIP user agent gets an emergency notification [25], it can control networked appliances for emergency handling.

Based on the investigation on the overlap and interaction among SIPC functions, we present the approaches on multi-function integration below.

3.4 Approaches for multi-function integration

The integration methods can be build-in and interprocess-control. The build-in method is to hardcode functions into a user agent so the user agent can invoke the functions by using API calls. The functions integrated by the build-in method are tightly coupled with each other. They can share code with each other and easily interact with each other by API calls. The interprocess-control method puts functions outside the user agent. The functions integrated by the interprocess-control method may interact with each other via interprocess communication, such as Dynamic Date Exchange (DDE [26]) and Message Bus (MBUS [27]). When adding a new function, three criteria may help to choose an appropriate integration method. First, the build-in method is more applicable if the new function shares many components with the existing functions. Second, if the

new function interacts with the existing functions extensively, the build-in method is preferable. Third, if there are existing popularly used applications supporting the new function, the interprocess-control method is more appropriate. Below we illustrate how we apply the criteria in integrating the function set of SIPC.

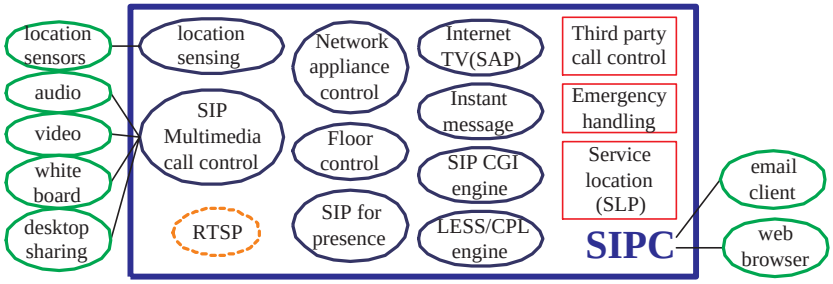

Fig. 4. Function set in sipc

Figure 4 shows the function set of SIPC. The functions inside the big thick-line-rectangle are integrated into the SIPC core, others are running in separate processes and controlled by the core by interprocess communication.

In SIPC's function set, all SIP related functions, such as SIP call setup, SIP DO method for networked appliance control, SIP event notification, SIP for instant messaging, SIP third-party call control, and SIP emergency call handling, share the same SIP stack and are tightly related to each other. These functions should be integrated in build-in way and put into one application. If we choose to use SIP and SOAP [28] for conference floor control [19], the floor control function should also be put into the same application.

To support the Session Announcement Protocol (SAP) and the Real Time Streaming Protocol (RTSP), based on the investigation in the Section 3.2 and 3.3, and the service examples in Section 2, we consider the best way is to integrate them in build-in way with the SIP functions. Both SAP and RTSP sessions use SDP for session description, and RTP for multimedia transmission, the same as SIP multimedia sessions, so code sharing is possible. Using external SAP and RTSP applications requires communication interfaces between SIP functions and SAP and RTSP functions. The communication interface is not trivial to build to handle function interactions.

The Service Location Protocol (SLP) support can be either build-in or interprocess-control because there is not much code sharing between SLP support and other functions. The communication interface between an SLP client and a SIP user agent can be simple. We choose to build an SLP client into SIPC because the implementation effort is not much but it is easier to perform function interactions.

There are two modes for location information retrieval. In the first, a user agent determines its own location, and announces it to other system components that need the information. We name this active location sensing. For example, the user agent

can use GPS or measure the field strength of wireless access points [29] to get the location information. Active location sensing may involve different kinds of location sensors. Instead of building all the location sensing technologies into SIPC, we use the interprocess-control method to integrate active location sensing functions. When SIPC starts, it listens on a TCP port for location information. Location sensors can send location documents in GEOPRIV Location Object Format [12] to that port.

The other mode is passive location sensing. In passive mode, a user's profile is put in a small device, such as an IR/RF programmable badge or an i-Button [30]. The device reader in a context can read the user's profile and send the information to a location server. The user needs to subscribe to the location server to get his own location information. SIPC implements the SIP event notification architecture [20] to handle location subscriptions and notifications.

In terms of function support for email and web browsing, we noticed that there are many existing email and web browsing applications. Instead of implementing email and web functions into SIPC, the preferable way is to integrate email and web browsing functions in the interprocess-control way. For example, on a Windows platform, by setting proper Windows Registry values, people can invoke SIPC from a web browser, or invoke a web browser from SIPC.

4 Program multi-function interactions

In Section 2, we presented some new services but without describing how to perform these services. Instead of hardcoding these services one by one, it is more convenient to make the services programmable by service scripts and customizable. We defined a service creation script language named Language for End System Services (LESS) [3]. LESS is extended from the Call Processing Language (CPL) [23], but with more emphasis on end system service creation. We choose to use LESS as the service creation language for SIPC because it is designed to be simple, easy to understand, and safe for end users to use. The simplicity and the tree-like structure of LESS make the graphical description of a LESS script and its XML document fully exchangeable. Any valid LESS scripts can be converted into graphical decision trees, and vice-versa. Though general purpose programming languages, such as C/C++ and Java, may also have graphical development environment, a graphical interface of an arbitrary program written in C/C++ or Java is extremely unlikely to be able to do anything more than represent the language constructs in the most basic manner. We did an in-depth analysis of the simplicity and safety of LESS [31] and developed a graphical service creation environment (SCE) for LESS, which is presented in Section 5.4.

The script below shows a LESS service script performing stereo control based on the caller's address. With the script, if the call is from `sip:boss@example.com`, the script will turn off the stereo.

```
<less>
  <incoming>
    <address-switch field="origin">
      <address is="sip:boss@example.com">
        <device:turnoff device="sip:stereo@room1.example.com"/>
```

```
      </address>
    </address-switch>
  </incoming>
</less>
```

A more complicated LESS service script below requires the integration of presence information handling, location sensing and instant messaging. When a SIPC instance equipped with the script receives an event notification showing that Bob, whose SIP URI is `sip:bob@example.com`, is online, it will check the location relation between the script owner and Bob. If they are at the same floor and close to each other, the script generates an instant message to Bob.

```
<less>
  <EVENT:notification>
    <address-switch> <address is="sip:bob@example.com">
      <EVENT:event-switch> <EVENT:event is="open">
        <location-relation-switch uri1="sip:bob@example.com">
          <location-relation distance="10" same="FLR">
            <location url="sip:bob@example.com">
              <IM:im message="Hi, I'm next to you"/>
            </location>
          </location-relation>
        </location-relation-switch>
      </EVENT:event> </EVENT:event-switch>
    </address> </address-switch>
  </EVENT:notification>
</less>
```

To incorporate all the new functions, we need to extend LESS with new packages, such as networked appliance control, presence information handling, SAP session handling, and location information handling. For example, in the above service scripts, we have the action `device:turnon` defined for networked appliance control. Defining a new LESS package is covered in [3] and [23].

5 Implementation

SIPC [2] is a SIP user agent written in Tcl/Tk and C/C++. It is originally developed to handle Internet telephony calls. Figure 4 shows SIPC's function set. The functions circled in solid have already been implemented, the functions circled in dotted line are under development, the functions in rectangle are partially implemented. As we integrated more and more functions into SIPC, we found many new services introduced by multi-function integration, some examples as we described in Section 2. Below, we briefly introduce the main user interface, the service creation environment, the functions for location sensing, and emergency call handling in SIPC.

5.1 The main user interface of SIPC

Figure 5 shows the main user interface of SIPC. By clicking on different function buttons, users can manually invoke different functions. In the service frame, a user can

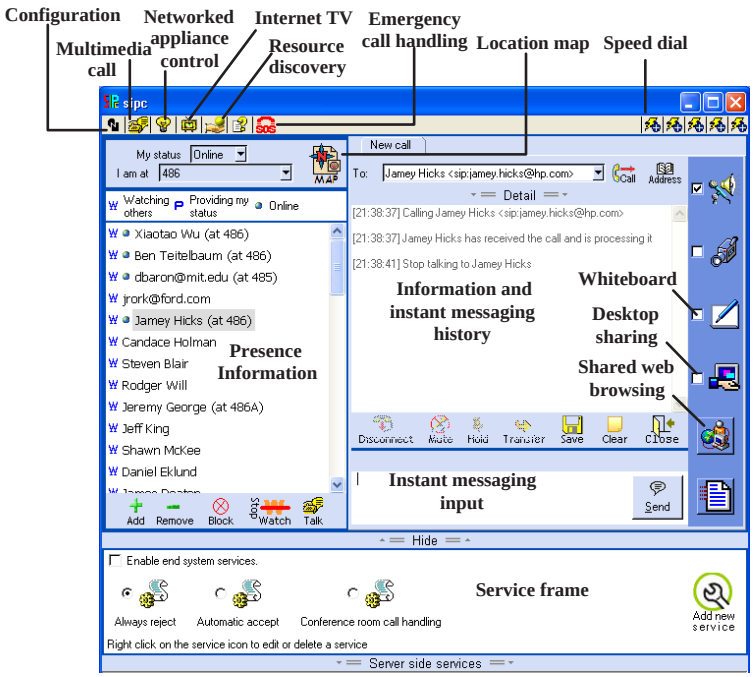

Fig. 5. Main user interface of SIPC

program new services to automatically handle multi-function interactions. We detail the service creation environment of SIPC in Section 5.4. In the 'presence information' frame, the user can see not only the buddies' presence status, but also their locations. If the user click the 'location map' button, the buddies' locations can be pinpointed in location maps.

5.2 Location sensing and location-based services in SIPC

Figure 6 shows the location map in SIPC. Buddies' locations are pinpointed on the map. In the map, a room can be a communication target. As shown in the figure, right click on a room, a user can easily broadcast instant messages or conference call invitations to all his buddies in the room.

As shown in Figure 7, SIPC supports both active location sensing and passive location sensing. In active mode, SIPC can get its civil location encoded in DHCP options for civic addresses [21] from a DHCP server, or get its geospatial location by reading the serial port connected to a GPS receiver. It can also get its location information from the Bluetooth beacon sent by a location server. In passive mode, users can use iButton,

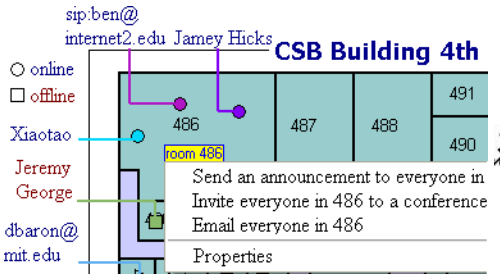

Fig. 6. Location map in SIPC

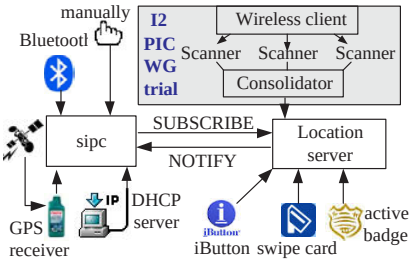

Fig. 7. Location sensing in SIPC

swipe card (such as university ID card), or active badge to generate location information and store the location in our location server. The location server can then NOTIFY SIPC about the location changes.

SIPC was used by the Internet2 Presence Integrated Communication working group (PIC WG) for their rich presence trials in Internet2 member meeting in Fall 2003 and Spring 2004. SIPC used passive location sensing in the trials. The gray area in Figure 7 shows the trial setup. Multiple scanners gathered signal strength from the wireless client on which SIPC was running. The consolidator calculated the location based on the signal strength information and sent it to the location server. The location server sent the location information to SIPC by SIP NOTIFY requests.

5.3 SIPC for emergency call handling

Figure 8 shows the emergency call handling architecture we are developing using SIPC. At the caller side, SIPC acquires its location, e.g., from DHCP options. Since different countries may have different emergency numbers, SIPC will send a NAPTR [32] request to the DNS server to get the local emergency numbers. When a user dials a number, SIPC will check whether it is an emergency number or not. If it is an emergency call, SIPC will encode the location in MIME in the outgoing SIP INVITE request. At the emergency call taker side, SIPC can pinpoint the caller on a map based on the location encapsulated in the INVITE request. In the figure, the SIP proxy server helps to route emergency calls to an appropriate emergency communications center.

5.4 Service creation environment in SIPC

We have developed a graphical service creation environment for SIPC. As shown in Figure 9, a user can simply drag a trigger events, such as *incoming* call, into the drawing area, then put different switches, such as *location-switch* and *address-switch*, for condition matching, then put different actions, such as *accept* or *reject* a call, into the drawing area. The user can connect these elements into a decision tree. The decision tree can be translated into a LESS script. SIPC can then handle calls based on the LESS script.

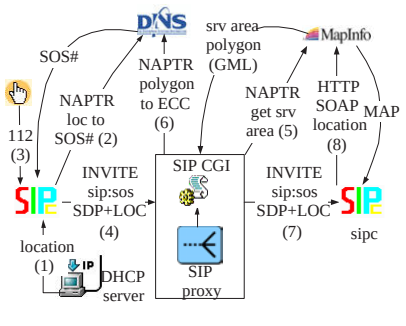

Fig. 8. Emergency call handling using SIPC

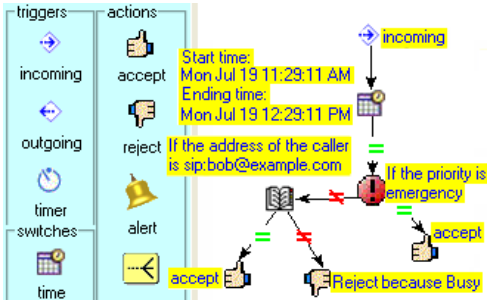

Fig. 9. Service creation environment in SIPC

6 Conclusion and future work

In this paper, we described how to integrate multiple Internet-oriented functions in our SIP user agent, SIPC, and presented the new services facilitated by the multi-function integration. The integration is not simply putting all the functions together but run them separately, instead, a careful design is required to minimize the overall complexity of the application, and enable function interactions. Multi-function interactions enable many innovative services that are otherwise impossible. We use LESS service scripts to automate the interactions. We also briefly introduced the implementation details of SIPC. We will investigate more Internet services, such as conference control, quality of service handling, and user profile management, for integration and define more LESS packages for new services.

References

1. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A.R., Peterson, J., Sparks, R., Handley, M., Schooler, E.: SIP: session initiation protocol. RFC 3261, Internet Engineering Task Force (2002)
2. Wu, X.: (Columbia university SIP user agent (sipc)) http://www.cs.columbia.edu/IRT/sipc.
3. Wu, X., Schulzrinne, H.: Programmable end system services using SIP. In: Conference Record of the International Conference on Communications (ICC). (2003)
4. Nirvis Inc.: (Slink-e) http://www.nirvis.com/slink-e.htm.
5. Moyer, S., Maples, D., Tsang, S.: A protocol for wide-area secure networked appliance communication. IEEE Communications Magazine **39** (2001) 52–59
6. Guttman, E., Perkins, C.E., Veizades, J., Day, M.: Service location protocol, version 2. RFC 2608, Internet Engineering Task Force (1999)
7. Rosenberg, J., Peterson, J., Schulzrinne, H., Camarillo, G.: Best current practices for third party call control (3pcc) in the session initiation protocol (SIP). RFC 3725, Internet Engineering Task Force (2004)
8. Berger, S., Schulzrinne, H., Sidiroglou, S., Wu, X.: Ubiquitous computing using SIP. In: ACM NOSSDAV 2003. (2003)

9. Shacham, R., Schulzrinne, H., Kellerer, W., Thakolsri, S.: An architecture for location-based service mobility using the SIP event model. In: Mobisys Workshop on Context Awareness. (2004)
10. Borenstein, N., Freed, N.: MIME (multipurpose Internet mail extensions) part one: Mechanisms for specifying and describing the format of Internet message bodies. RFC 1521, Internet Engineering Task Force (1993)
11. Schulzrinne, H.: RPID – rich presence information data format. Internet draft, Internet Engineering Task Force (2003) Work in progress.
12. Peterson, J.: A presence-based GEOPRIV location object format. Internet Draft draft-ietf-geopriv-pidf-lo-01, Internet Engineering Task Force (2004) Work in progress.
13. Niemi, A.: Session initiation protocol (SIP) extension for event state publication. Internet Draft draft-ietf-sip-publish-02, Internet Engineering Task Force (2004) Work in progress.
14. Handley, M., Perkins, C.E., Whelan, E.: Session announcement protocol. RFC 2974, Internet Engineering Task Force (2000)
15. Handley, M., Jacobson, V.: SDP: session description protocol. RFC 2327, Internet Engineering Task Force (1998)
16. Mahy, R.: A message summary and message waiting indication event package for the session initiation protocol (SIP). Internet draft, Internet Engineering Task Force (2003) Work in progress.
17. Fielding, R., Gettys, J., Mogul, J.C., Frystyk, H., Berners-Lee, T.: Hypertext transfer protocol – HTTP/1.1. RFC 2068, Internet Engineering Task Force (1997)
18. Schulzrinne, H., Rao, A., Lanphier, R.: Real time streaming protocol (RTSP). RFC 2326, Internet Engineering Task Force (1998)
19. Schulzrinne, H., Wu, X., Koskelainen, P., Ott, J.: Requirements for floor control protocol. Internet Draft draft-ietf-xcon-floor-control-req-00, Internet Engineering Task Force (2004) Work in progress.
20. Roach, A.B.: Session initiation protocol (sip)-specific event notification. RFC 3265, Internet Engineering Task Force (2002)
21. Schulzrinne, H.: DHCP option for civil location. Internet draft, Internet Engineering Task Force (2003) Work in progress.
22. Lennox, J., Schulzrinne, H., Rosenberg, J.: Common gateway interface for SIP. RFC 3050, Internet Engineering Task Force (2001)
23. Lennox, J., Wu, X., Schulzrinne, H.: CPL: a language for user control of Internet telephony services. Internet draft, Internet Engineering Task Force (2003) Work in progress.
24. Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V.: RTP: a transport protocol for real-time applications. RFC 3550, Internet Engineering Task Force (2003)
25. Schulzrinne, H., Arabshian, K.: Providing emergency services in Internet telephony. IEEE Internet Computing **6** (2002) 39–47
26. Netscape corporation: (Netscape's DDE implementation) http://developer.netscape.com/docs/manuals/ communicator/DDE/index.htm.
27. Ott, J., Perkins, C.E., Kutscher, D.: A message bus for local coordination. RFC 3259, Internet Engineering Task Force (2002)
28. World Wide Web Consortium: (Simple object access protocol (soap) 1.1) http://www.w3.org/TR/SOAP/.
29. Niculescu, D., Nath, B.: Ad hoc positioning system (APS). In: GLOBECOM (1). (2001) 2926–2931
30. Dallas Semiconductor: (iButton) http://www.ibutton.com.
31. Wu, X., Schulzrinne, H.: The simplicity and safety of the language for end system services (LESS). Technical report, Department of Computer Science, Columbia University (2004)
32. Mealling, M., Daniel, R.W.: The naming authority pointer (NAPTR) DNS resource record. RFC 2915, Internet Engineering Task Force (2000)