A Generic Event Notification System Using XML and SIP

Knarig Arabshian and Henning Schulzrinne
Department of Computer Science
Columbia University, New York, NY
{knarig,hgs}@cs.columbia.edu

***Abstract*— We discuss a generic event notification system using XML and SIP. XML messages are used for configuration and filtering of events as well as the generation of GUI. Event notification handling is implemented using SUBSCRIBE and NOTIFY methods. The integration of XML and SIP offers greater flexibility in creating a robust event notification architecture. We describe the protocol design and implementation of such a system.**

I. Introduction

Event notification systems span a broad range of applications such as in healthcare, business or the government. Since the underlying mechanism of subscriptions and notifications is the same, a system needs to exist which can be applied to any field. Integrating SIP and XML allows for a generic event notification system that can be applied in any field. In Section II we give a brief overview of SIP and describe its publish/subscribe/notify mechanism. In Section III, we describe how XML can be incorporated into the event notification system to configure the types of events supported to automatically generate a graphical user interface for subscription and filter out events within a subscription. Finally, an implementation is discussed in Section IV.

II. Overview of SIP

SIP [1] is part of the IETF standards process and is similar to other Internet protocols such as SMTP (Simple Mail Transfer Protocol) used for e-mail and HTTP (Hypertext Transfer Protocol) used for the World Wide Web. It is a text-based protocol that is used to manage sessions established on the Internet. These sessions can either be simple two-way telephone calls or collaborative multi-media conference sessions. Once the session has been set up using SIP, the audio and video packets are transported using RTP (Real-time Transport Protocol). The underlying transport of SIP messages can either be in UDP, TCP or Stream Control Transmission Protocol (SCTP). Within a multimedia conference, the body of a SIP message often contains a session description which enumerates the media streams to be used for the session. However, as we describe below, the message body can contain any other type of information which is relevant to the current use of SIP.

SIP has also been extended to generate event notifications and instant messages [2] [3]. Users subscribe to an event with the SUBSCRIBE method and receive notifications via NOTIFY. This event notification facility is used for events that occur during telephone calls for presence notification and also may be used within generic SIP event notification systems. A user agent client sends a SUBSCRIBE request to the appropriate server. This request contains an "Event" header indicating the type of event the user is subscribing to. In the case where a user is interested in a number of events, it sends multiple SUBSCRIBE messages. The request's "Expires" header specifies the duration of the subscription. SUBSCRIBE messages can be refreshed whenever the subscription has expired–if a subscriber wants to unsubscribe, it can send a SUBSCRIBE message with an expiration time of zero. Once the server receives the subscription, it adds the subscriber to the appropriate event list and approves of the subscription and then generates NOTIFY requests to the subscriber when an event occurs. When a device is either no longer interested or capable of receiving events, subscriptions time out in order to prevent the devices from consuming network resources.

Another feature of SIP is that it supports a method, which is used to control appliances connected to the computer. This capability allows SIP to control electric devices remotely. Thus, when issuing a NOTIFY, remote procedure calls (as described below) can be made to the suscriber's system which automatically invokes a function on a connected device.

Security and confidentiality of information passed within event notification systems is crucial. SIP can use existing security mechanisms like HTTP Digest or transport layer security (TLS). HTTP Digest mechanism is used for authentication using a shared password, but

it does not provide encryption of the messages.

III. Incorporating XML

In our system, the event notification server maintains XML schemas describing the events supported. An event notification server may get its information from many different sources such as weather alert centers or communication-related events, etc. Each of these institutions will provide the server with an XML schema describing the events it supports. For example, the weather alert center may provide a schema which illustrates the various weather alerts it supports such as rain or blizzard warnings. Within each of these subevents, the schema may also specify different parameters the user can subscribe to. For instance within the rain subevent, a parameter may be specified which indicates the percentage of the chance of rain needed before notifying the subscriber. Thus, if a user subscribes to the rain alert, it may only want to be notified if there is a good chance of rain. The schema can be fine tuned to include parameters which filter out the events and subevents in order to create a notification service which adheres to the user's preferences. The schema can also configure the types of alerting methods offered by the notification server and the remote procedure calls it can make to the client. Different methods of alerting can be anything that pertains to the situation such as being paged, flashing lights or sounding alarms.

XML schemas can further automate the subscription process by allowing the client to automatically generate a graphical user interface for the subscriber to input his information, creating a more user-friendly application. Thus, the client can subscribe to any notification server, obtain the XML schema, and create the GUI which correlates to the schema. This affords greater flexibility and allows the client to participate in any type of event notification service that supports this system.

While XML schemas are used for configuration of events and automatic GUI generation, XML messages within the SIP SUBSCRIBE body are also used for filtering out the subscription of the event on the client side. Filtering events using XML is now being discussed within the IETF [5]. The ”Events” header in a SIP SUBSCRIBE message specifies an event package to subscribe to, but does not specify the event filters. Thus, the information that the user inputs in the GUI form is packaged in an XML document within the SIP SUBSCRIBE message body and sent to the notification server for detailed monitoring.

In addition to using XML for subscription, it can also be used to enhance notification via remote procedure calls. SOAP (Simple Object Access Protocol) [6] is an XML-based remote procedure calling mechanism. It facilitates a program running in one platform to communicate with a program within the same or remote platform. It is often used with HTTP, but can also be used with any other application transport protocol such as SMTP or in this case SIP. The fact that SOAP is interoperable between different platforms, makes it very convenient to use for a generic system. When the notification server receives the XML filter within the subscribe message, it processes the alert methods the user has subscribed for. Thus, when the notification server sends a NOTIFY, it embeds a SOAP message within the body that represents a call to a function on the subscriber's system. The SOAP message is then processed and the function call is executed accordingly.

IV. Protocol Implementation

We are currently implementing a prototype version of a generic SIP-based event notification system which we are applying to emergency notification [7] and medical event monitoring [8]. The Columbia SIP user agent, sipc, and the SIP proxy server, sipd, are being extended to handle event notifications using XML messages for configuring, filtering and remote procedure calls. Below we will describe the flow of our implementation.

Initially, once the user runs sipc, an empty SUBSCRIBE message is sent to the event notification server (sipd). Sipd sends back a message with an XML document that lists the different events it supports. Sipc displays these events to the user and when the user clicks on one of them, sipc sends out another SUBSCRIBE indicating the event in the ”Events” header. Sipd then sends back a reference to an XML schema that describes the configuration of that event. Sipc processes this schema by recursively going through each event and saving its elements and their types in a recursive list. It then goes through the list and displays each element in a graphical interface that is similar to a form that must be filled out. Thus, for every element, it first checks its type, such as string or decimal, and then displays it in the form accordingly. When the user completes the information requested in the form, sipc packages this in an XML filter and sends it out within another SUBSCRIBE message to sipd. Sipd recognizes this message as a filtered subscribe, adds the user to its database and saves the user's subscription preferences. Once sipd receives an event notification from one of its event sensors, it checks to see which of its users fits in the category of that event notification and sends out a NOTIFY message to the interested parties. Figure 1 illustrates the protocol flow.

We are currently developing the server side. Sipd will receive PUBLISH messages from various event sensors and will save the different types of events it supports.

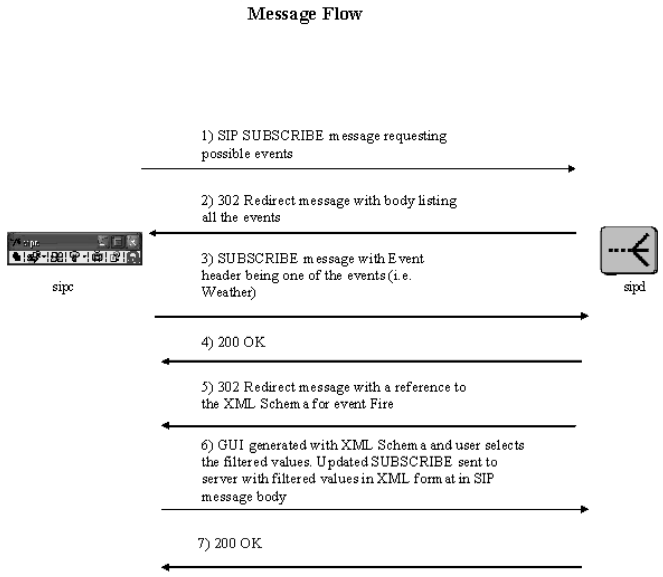

Fig. 1. Protocol exchanges for event alerting

V. Benefits

There are a number of benefits of implementing an event notification system that uses SIP and XML.

***Device neutrality*:** There are likely to be range of SIP-based end systems, ranging from IP telephones, 3G wireless handsets, IM/presence software to embedded devices. Thus, the event notification system can migrate to new devices, without having to be extended.

***Generic in its application*:** Since we are using XML schemas and messages to configure and filter the events, this system can be applied to any type of event notification. As long as a schema is created to configure the events and all the clients and servers are aware of this schema, the notification system can be used for a variety of applications.

***More information*:** The SIP event notifications can carry much more detailed information, tailored to different needs. For example, automated systems cannot benefit from voice announcements, but need detailed information on duration of events and the ability to cancel alerts. Even for human users, it becomes much easier to provide a multitude of languages.

***Automated action*:** Related to the previous items, SIP event notifications can provide more detailed guidance. For example, the message can contain pointers to web pages providing detailed instructions on appropriate behavior. It can also administer remote procedure calls in events related to distributed simulation or games.

***Out-of-area notification*:** There are cases where the recipient of the information is currently away, but still needs to be alerted. For example, owners of summer cottages need to be aware of impending storms so that they can summon appropriate assistance or have their property checked on after the fact. Those at work may need to know about conditions affecting their home. Since alerting methods are flexible, there is no need to be in close physical proximity of the event in order to be notified of it.

VI. Conclusion

The widespread use of event notification systems creates the need for a generic system. We have discovered the outline of such a system.

References

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. R. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: session initiation protocol. RFC 3261, Internet Engineering Task Force, June 2002.

[2] A. B. Roach. Session initiation protocol (sip)-specific event notification. RFC 3265, Internet Engineering Task Force, June 2002.

[3] Session initiation protocol (SIP) extension for instant messaging. RFC 3428, Internet Engineering Task Force, Dec. 2002.

[4] D. C. Fallside. XML schema part 0: Primer. Technical report, May 2001.

[5] H. Khartabil et al. Event notification filtering for presence. Internet draft, Internet Engineering Task Force, Jan. 2003. Work in progress.

[6] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, S. Thatte, and D. Winer. Simple object access protocol. Technical report, World Wide Web Consortium W3C, May 2000.

[7] H. Schulzinne and K. Arabshian. Providing emergency services in Internet telephony. In *Society of Photo-Optical Instrumentation Engineers*, Boston, Massachusetts, Aug. 2002.

[8] K. Arabshian, H. Schulzrinne A SIP-based Medical Event Monitoring System In *5th International Workshop on Enterprise Networking and Computing in Healthcare Industry (HealthCom)*, Santa Monica, CA, June 2003.