Setting up Value-added Services

This chapter introduces the concept of added-value services and their implementation in IP Telephony products. Value-added services are services that integrate VoIP with other protocols and services. Seamless integration with Web, E-mail and potential other Internet services provides convenience which, is a key feature of Internet telephony.

6.1 Web integration of H.323 services

A natural extension to any Internet service nowadays is its integration with the Web. Unfortunately, H.323 is more difficult to integrate with Web interfaces than SIP, due to the complexity of the protocol and its roots in the telecom world. Web applications that would be much needed in the area are the following:

- **Presence**: Web-based applications that allow a group of users to indicate their availability for accepting calls and to check availability of others. Analogous to the classic ICQ-type application, H.323 presence applications could stir-up significantly more users if they were widely available;
- **Click-to-dial**: Web-based applications that allow users the ease to click in order to make a call to a target endpoint, service set up or user. Obviously, H.323 endpoint applications are harder to set up than e.g., MP3 client-applications that download media streams, and this has a limiting effect on their use. Achieving the ease of use of a click-to-dial interface would have a dramatic effect on the deployment of H.323;
- **IPPBX management**: Web-based applications that allow administrators of an IP-based PBX system to monitor and control calls being made. The simplicity of use of such Web interfaces would allow the equivalent of a switchboard operator to: transfer calls, initiate multi-party calls, answer calls on busy, and terminate calls.

The applications above are mostly found in commercially-available integrated solutions and they employ proprietary methods that are extremely difficult to integrate with custom-made Web interfaces. If you are interested in setting up your own Web-based application and integrate it with your H.323 services, you must find the means of connecting it to the gatekeeper you have deployed. The methods are many and vary from gatekeeper to gatekeeper, depending on the available APIs and supported interfaces.

In this section, we will simply list some of the options available for interfacing with common gatekeepers. For more information please refer to Appendix B.

6.1.1 RADIUS-based methods

If you are simply interested in making a presence application, then any gatekeeper with RADIUS support can be interfaced with, through proper RADIUS server set-up. For example, the
FreeRADIUS server can be configured to execute an external script each time an authentication is made. This method can be used to keep a list of currently registered endpoints on the gatekeeper and make it available over the Web as a primitive means of advertising availability. A slightly better method would be to configure FreeRADIUS with MySQL back-end and maintain a database table with currently registered endpoints, by updating the table for each RADIUS event (authentication and accounting).

### 6.1.2. SNMP-based methods

An alternative to the above method is to use SNMP to interface with your gatekeeper, assuming it has SNMP support. The Cisco MCM has SNMP support but with limited functionality, while the RADVISION ECS supports the H.341 standard for SNMP access. You can explore the options available to an external application, interfacing with SNMP by carefully studying the supported MIBs for your gatekeeper.

### 6.1.3 Cisco MCM GK API

The Cisco MCM Gatekeeper actually implements a very flexible API for receiving events that the administrator is interested in processing. The external application can choose to further process the event, or just log it for informational purposes. The API interface to the MCM is proprietary and based on the GKTMP ad-hoc protocol for informing the external application of **RRQ**, **URQ**, **ARQ**, **LRQ**, **DRQ**, **BRQ** and related (confirm/reject) events. The API allows the external application to issue respective confirm and reject (**xCf**, **xRJ**) commands for specific events, based on external application logic. Cisco provides a library in C and template code for an external application, but it is not fully working code and requires significant resources to be applied before a demo application can be built. Also commercial solutions from Cisco partners exist which are based on this same API (see GK API Guide Version 4.2).

### 6.1.4 GNU GK Status Interface

The GNU Gatekeeper provides a very useful command-line interface for monitoring and control of gatekeeper operations. It is called the ‘Status Interface’ and allows telnet connections from remote administrative nodes to connect and monitor **RCF/RJ**, **UCF/RJ**, **ACF/RJ**, **LCF/RJ**, **DCF/RJ**, **BCF/RJ** events with detailed information.

Additionally, it allows monitoring of call detail records (**CDR**) for accounting applications and **RouteRequest** messages for interfacing with the ‘Virtual Queues’ feature, proprietary to the GNU Gatekeeper. The fact that many different nodes can connect at the same time over this administrative interface and process different events of the gatekeeper, allows for a distributed and flexible implementation of monitoring services. Indeed, a number of tools have been developed that build on this interface and provide interesting functionality:

- **OpenH323 Gatekeeper Java GUI**: this interface allows the monitoring of registrations and calls on the gatekeeper and provides endpoint information as well. Source code is available to modify for added functionality, if needed;
- **Sample ACD application**: this interface allows the definition and management of groups of endpoints (agents) who will handle a large volume of calls for a single alias. The ACD will check which of the agents is qualified and available (not in another call and not logged off from ACD work) and informs the gatekeeper which agent will receive the call. If no agent is available, the ACD will tell the gatekeeper to reject the call. All call routing logic is kept out of the gatekeeper to ensure stable operation, while routing logic can be changed frequently.

- **PHP GNUgk Status Monitor** - v0.4: this application allows monitoring of registered endpoints and calls in progress through a PHP Web interface. Call disconnection is possible and further functionality is being developed. Source code is available.

### 6.2 Web integration of SIP services

This chapter provides an overview of some added-value services implemented using a Web interface. Serweb, the Web interface for the SIP Express Router described in Chapter 4, will be used as an example implementation of Web-based added-value services for SIP.

Integration of SIP services with Web interface is the most common scenario. A Web interface is often used for the provisioning of SIP products and for the implementation of advanced services. Web browsers are available in the vast majority of existing operating systems.

#### 6.2.1 Click-to-dial

Click-to-dial is a method of establishing a call between participants using a Web interface. It greatly simplifies dialling, in that calling parties do not have to dial lengthy addresses and they keep their phonebooks separately from SIP phones. In its simplest form, a user has a Webpage where he can enter the SIP addresses of two users, and the SIP user agents of those two users get connected. We will focus on REFER-based click-to-dial.

A REFER-based click-to-dial scenario is based on the paradigm of intelligent end-devices and dumb network. One of the involved SIP user agents is asked to connect to the other and report to the server when it is done.

The drawback of this approach is that one of the involved SIP user agents must support the REFER SIP method which has been standardised recently (see RFC3515). The big advantage that balances the previous drawback is that it is extremely easy to implement REFER-based click-to-dial in the server. Call-flow for REFER-based click-to-dial is depicted in Figure 6.1.

First, the SIP server sends an INVITE to one of the phones because phones usually do not accept REFER without prior invitation. The INVITE contains 0.0.0.0 as the IP address in SDP, because there is no remote phone (the message is sent by user agent within the SIP server which does not deal with media).

After that, the server sends a REFER method which will ask the phone to send INVITE somewhere else. The URI of the called party is passed to the phone in a Refer-To header field of the REFER method.
The phone sends a **NOTIFY** method back to the server once the connection is established. The click-to-dial feature allows the creation of many advanced features, like phone-book, in which you can click on an entry and your phone and phone of the person represented by the entry are connected. You can implement a list of missed call in exactly the same way and clicking on an entry in the list will connect your phone with that person. There are many others possible scenarios.

Figure 6.1 **REFER**-based click-to-dial

### 6.2.2 Presence

It is also possible to display the presence of SIP users in a Webpage (for example, in a phone-book). Displaying the on-line status of subscribers allows calling parties to determine availability and willingness to have a conversation conveniently. The status may be shown, for example, in the calling party’s phonebook or on the called party’s homepage. Linking on-line users to click-to-dial applications greatly integrates telephony with the Web and introduces convenience to users.

When a SIP phone registers, the SIP server records this information into a database. The Web server can then access the database to see if a user is online or offline.

Go to [iptel.org](http://iptel.org), create a new account, and insert some entries into your phone-book to see how it works.
6.2.3 Missed calls

'Missed calls' is a feature that allows a user to display the list of call attempts that were made during the period when he was not online (registered in SIP). The list can be presented on a Web page.

Recording missed calls has a lot in common with accounting. Servers doing accounting log some information when a call is successfully established and torn down. When a calling party gets a negative final response to his call, or does not receive any reply at all (timeout), then the server also records this event with a flag, indicating that it was a missed call. This information can be later used to compile the list of missed calls.

6.2.4 Serweb

Serweb is a Web front-end for a SIP Express Router (see Section 4.6.2 for more details). Serweb creates a user interface for users of the proxy server, where they can manage their account, change their configuration and do many advanced things.

6.2.4.1 Installation

Serweb is a set of php scripts. To run it, you will need Apache Web server with php and mysql support. Because a SIP Express Router and serweb talk together using a FIFO interface, the SIP Proxy and the Web server must be running on the same machine.

Get serweb from http://developer.berlios.de/projects/serweb and untar the archive. It is recommended not to untar it to the document root of your Web server. Alternatively you can get serweb using CVS:

```bash
export CVSROOT=:pserver:anonymous@cvs.berlios.de:/cvsroot/serweb
cvs login
cvs co iptel
```

6.2.4.2 Configuration

The entire configuration of serweb is in config.php file in html subdirectory. You will need to configure the following:
- Host on which MySQL server is running: $this->db_host="localhost";
- Path of the user interface on the Web server: $this->root_path="/";
- Root URI of the Web server: $this->root_uri="http://www.foobar..."
- Path of serweb images on the Web server: $this->img_src_path = $this->root_path."iptel_img/";
- Path of java script files of serweb: $this->js_src_path = $this->root_path."iptel_js/";
- Path of ccs files of serweb: $this->style_src_path = $this->root_path."iptel_styles/";
It is necessary to create some aliases in the configuration file of Apache Web server:

```bash
Alias /iptel_img "'/var/www/iptel/html/img"
Alias /iptel_styles "'/var/www/iptel/html/styles"
Alias /user "'/var/www/iptel/html/user_interface"
Alias /admin "'/var/www/iptel/html/admin"
```

Do not forget to update the directory path according to your real settings and make sure that you have `register_globals` and `short_open_tag` set to `On` in your `php.ini` file.

### 6.2.4.3 Operation

To login into serweb open: `http://<your_server>/user` in your Web browser. You will be prompted for username and password. The username and password is same as the one you are using in your SIP user agent to register at the server.

![Figure 6.2 Serweb - My Account](image)
The **My Account** tab (see Figure 6.2) allows users to change their preferences and modified registered contacts. They can also see aliases they created and permissions for calling to the PSTN.

Users can also create their own phone book (see Figure 6.3). In the phonebook, you can see the presence status of each user. If the user is currently registered, then you will see **online** in the status column. If he is not registered then you will see **offline** and if the user does not belong to administrative domain of the server, then you will see **non-local**.

![Figure 6.3 Serweb - Phonebook](image)

Clicking on the address of a user will establish a phone call between your and his phone, provided that your phone supports **REFER**, as described in Section 6.2.1.

The **missed calls plane** (see Figure 6.4), allows users’ to see their missed calls. Again, clicking on an entry will connect you with that **user** and **status** describes presence of the user as described in the previous section.
When anyone sends a **SIMPLE** message to a user that is currently not online, the server will store the message and send it later when the recipient comes online. In the plane **message store** (see Figure 6.5), you can see all messages that are stored for you.

### 6.2.5 SIP Express Router message store

Message store has been implemented as a separate module for a SIP Express Router. To use the module, you will need to load the module:

```
loadmodule "/usr/local/lib/ser/modules/msilo.so"
```

Configure the address of the server (it will be used when sending stored messages) and the URL of the database:

```
modparam("msilo", "registrar", "sip:registrar@<your_domain>")
modparam("msilo", "db_url", "sql://ser:passwd@dbhost/ser")
```
Then, when the server receives a `MESSAGE` request and it cannot deliver it because the recipient is offline, it will save the message:

```java
if (!lookup("location")) {
    if (method == "MESSAGE") {
        if (!t_newtran()) {
            sl_reply_error();
            break;
        }
        if (m_store("0")) {
            t_reply("202", "Accepted for Later Delivery");
            break;
        }
    }
    t_reply("503", "Service Unavailable");
    break;
}
```

When the `lookup` of recipient's location fails (the recipient is not registered), a new transaction is created (needed for `msilo` module). Save the message using the `m_store` command, and reply with `202 accepted`.
Each time we call `save(location)`, we have to check if the previously available user is registered again and if so, then send the stored messages. The following example shows how to do that:

```c
if (!save("location")) {
    sl_reply_error();
}

m_dump();
```

Command `m_dump` checks if the registering user has any stored messages and if so, sends them.

### 6.3 Voicemail

Another Internet application which lends itself for integration with telephony is e-mail. A traditional PSTN application, which can be replaced with VoIP and e-mail, is voicemail.

Traditional PSTN voicemail systems feature fairly inconvenient user interface for message retrieval: IVR (Interactive Voice Responder). Calling parties have to navigate through an automated voice menu, listen to lengthy announcements, type digits as prompted and be very patient to achieve very simple tasks. It is undoubtedly more convenient to deliver recorded messages to the called to party by e-mail. The called party can then listen, store and process the received messages at his convenience. The following picture shows the data flow in a voicemail-to-e-mail scenario. An open-source voicemail-to-e-mail application, SEMS, is available from iptel.org.

SEMS stands for SIP Express Media Server. It is an application framework that offers easily-built applications dealing with media streams. The framework itself provides only minimal functionality for accessing and manipulation of media streams and signalling. High-level logic is stored in additional modules that can be dynamically loaded.

Examples of such modules are voicemail, announcement server and ISDN gateway. Some other voicemail systems exist including OpenAM, which is available from the OpenH323 Website and a voicemail system built-in inside the VOCAL system.

Unfortunately, they are not easy to set up and they are not yet ready to be used in a production environment if you need a completely integrated product. They do not work without bugs and they can easily be customised for small environment scenarios.
VoiceMail2Email Example

Figure 6.6 Voicemail