



IP Communications Applications for Business Advantage

Metreos CallMonitor Administration Guide

Metreos Communications Environment 2.1

Copyright © 2005 Metreos Corporation
All Rights Reserved

Proprietary and Confidential Information

Table of Contents

Configuring the CallMonitor Application	3
SPAN Port	3
Metreos Application Runtime Engine Configuration.....	3
Telephony Server Creation.....	3
CallMonitor Application Configuration	3
CallMonitor Application Partition-Specific Configuration	5
CallManager Configuration.....	6
Mapping DID (Direct-Inward-Dial) Numbers	6
Configuring the PCap Service	8
Configuring the PCAP Service.....	8
Choose a Network Interface to Monitor	8
Configure the Network Adapter	8
CallMonitor Reporting	10
Using the Reporting Tool	10
Usage Reporting Database Schema.....	11
XML Schema	12

Configuring the CallMonitor Application

The Metreos CallMonitor® application for the Metreos Application Runtime Engine provides the means to randomly monitor a conversation on a phone selected from a group of Cisco IP phones, given that the IP traffic of the phones in question is routed to the Metreos Application Runtime Engine using a SPAN port associated with the Metreos PCap Service®.

The administrator of the Metreos environment can segment phones by directory number into distinct groups, designating a unique DID (Direct-Inward-Dial) number for each such group.

SPAN Port

A SPAN port must exist which is able to see all SCCP and audio traffic to and from the Cisco IP phones to be monitored.

Details on configuring a SPAN port are beyond the scope of this document.

Metreos Application Runtime Engine Configuration

Telephony Server Creation

The Metreos Application Runtime Environment must be configured to recognize any and all CallManagers hosting the Cisco IP phones eligible for monitoring.

Using the Metreos Management Console (<http://<Application Server IP>/mceadmin>), the administrator will create two Telephony Servers. Once logged into the Management Console, navigate to the *Telephony Servers* link. From there, create both a CallManager telephony server, and an H.323 telephony server. *The IP address of the H.323 telephony server should correlate to a Cisco CallManager which handles calls, this typically being a subscriber configured as part of a Cisco CallManager cluster.*

An alternate configuration scenario is one in which JTAPI is used (rather than H.323) as the signaling protocol between the PSTN and CallManager. Such a configuration is not described in this document..

CallMonitor Application Configuration

To access CallMonitor application configuration options, navigate to <http://<Application Server IP>/mceadmin/>, and from there select *Applications*. From the list of applications select the *CallMonitor* application. Configure the application as described below.

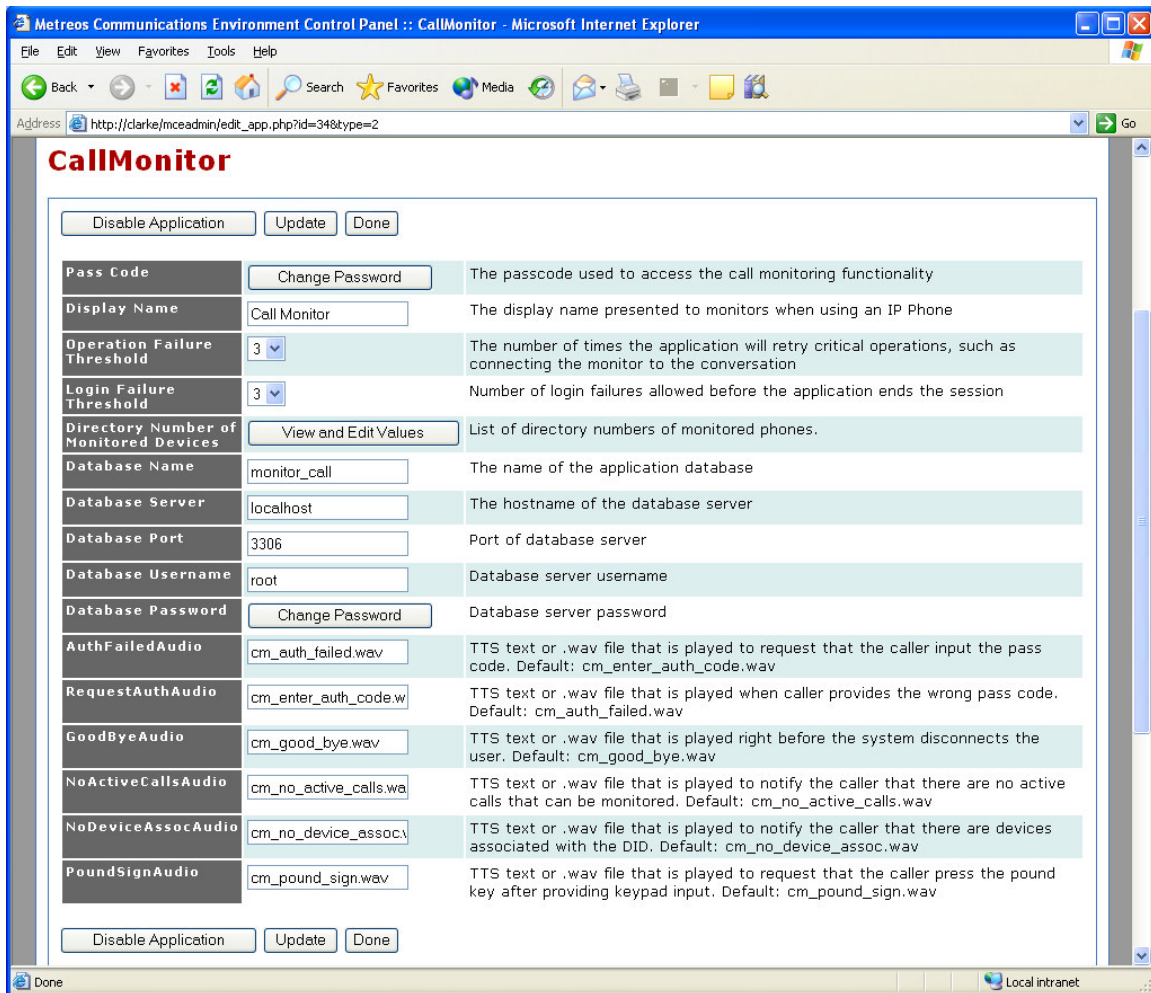


Figure 1: CallMonitor Application Configuration

Pass Code

The PIN code that an administrator must enter in order to monitor a conversation.

Display Name

The name presented to the monitoring user when an IP Phone is used to initiate the monitoring functionality.

Operation Failure Threshold

The number of times certain application operations will be attempted, such as bridging the monitoring user into the conversation.

Login Failure Threshold

The number of times an invalid or unrecognized PIN code may be entered

Directory Numbers of Monitored Devices

A list of directory numbers of the phones to be associated with a monitor group.

Other fields need rarely be modified.

CallMonitor Application Partition-Specific Configuration

Applications in the Metreos Application Runtime Engine can be *partitioned*, or split into mirrored instances, each having its own separate configuration. In addition to the configuration items listed above, each application partition can have its own set of configurable triggering criteria.

Configuring a Partitioned Application: An example of how and why one might use partitions with Metreos CallMonitor is as follows. An administrator creates two partitions for the CallMonitor application. The first partition monitors four directory numbers by having those four numbers configured in the Directory Numbers of Monitored Devices field, with the other partition monitoring six different numbers. The administrator might then set up the first partition to trigger when a directory number of 2000 is dialed, and the second partition to trigger on the number 2001,

The administrator could then configure unique pin numbers for each partition, and inform all parties interested in monitoring the two groups of phones of the extensions and authorizing PINs for each.

The specific steps you would take to configure this example are as follows:

1. Create a second partition, in addition to the default partition already existing.
 - a. Navigate to Mceadmin → Applications → CallMonitor → Create Partition
 - b. In the new partition page displayed, enable the partition and set the Media Resource Group to *Default*. The Call Route Group does not matter.
2. Configure the two partitions by establishing two groups of monitored devices, with unique PIN and triggering criteria. For each partition:
 - a. Configure a group of monitored devices in the partition:
 - i. Navigate to Mceadmin → Applications → CallMonitor → Edit [Partition] → View
 - ii. Edit the values for Directory Number of Monitored Devices.
 - iii. In the pop-up window that appears, enter the line number of each device for this partition.
 - b. Configure a PIN for the partition.
 - i. Navigate to Mceadmin → Applications → CallMonitor → Edit [Partition] → Change Password.
 - ii. In the popup window appearing, enter the old PIN and the new PIN. Monitoring agents must supply this PIN when using the application. Note that the first time the application is configured, the *Old Password* field should be left empty.
 - c. Configure the trigger parameter for the partition:
 - i. Navigate to Mceadmin → Applications => CallMonitor => Edit [Partition] => Edit Trigger Parameters [CallMonitor].
 - ii. Set the Parameter Name entry to “to”.
 - iii. Set the Initial Value entry to the routable directory number previously configured for the H.323 gateway.

CallManager Configuration

Mapping DID (Direct-Inward-Dial) Numbers

An outbound H.323 gateway should be created corresponding to the Metreos Application Server. When creating the H.323 gateway, the *Device Name* field should be the primary IP address of the Metreos Application Server.

Once the H.323 gateway is created, you will need to configure CallManager such that calls inbound from the PSTN can reach the newly created H.323 gateway corresponding to the Metreos Application Server. The most straightforward method of doing so is to configure a route pattern associated with the H.323 gateway. Any numbers reserved for DID access to the CallMonitor application should be taken into consideration during this step.

Figure 2 illustrates the call signaling paths involved in the CallMonitor application. On that diagram, note that the 'H.323 Interface' node on the Application Server is an outbound H.323 Gateway as far as CallManager is concerned. When a call comes in from the PSTN and is subsequently routed to the Application Server H.323 Gateway, the CallMonitor application will initiate.

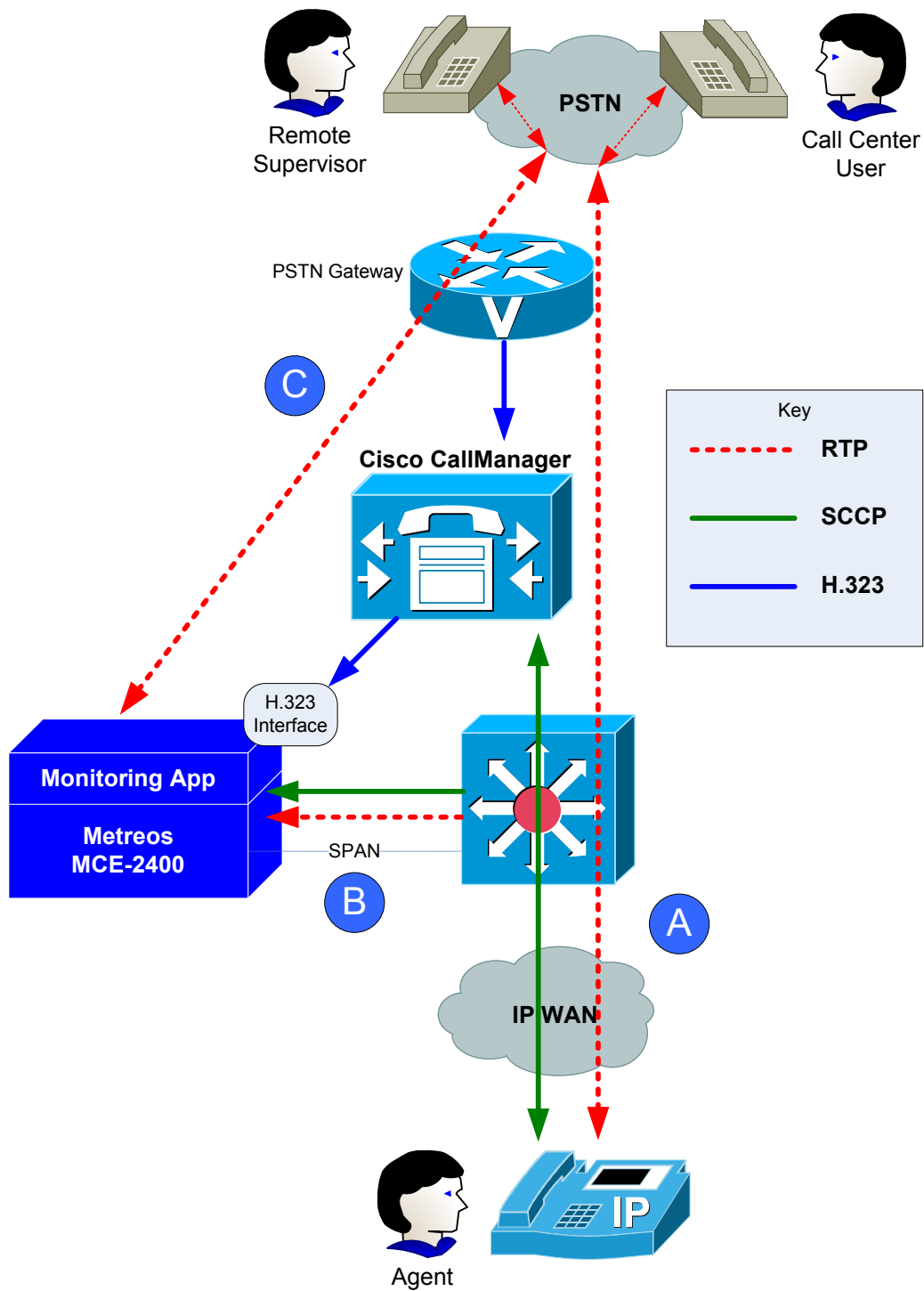


Figure 2: Remote Monitoring Block Diagram

Configuring the PCap Service

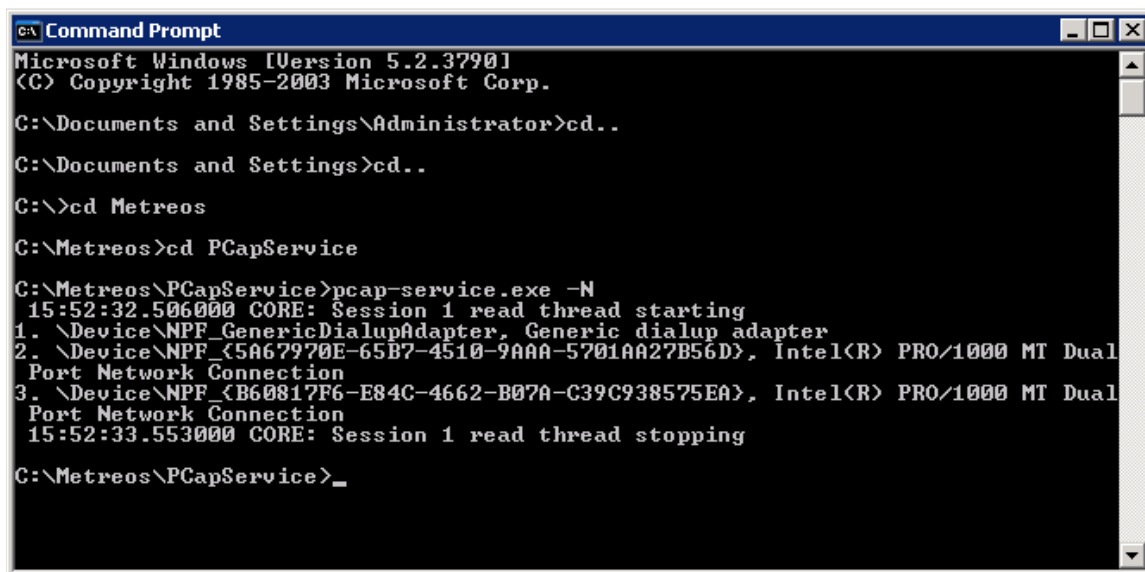
Configuring the PCAP Service

The PCAP network monitoring service must be configured to listen to a single network interface.

Two steps are required in order to configure the PCap Service to begin monitoring a specific network adapter.

1. Choose a Network Interface to Monitor

In a command window, execute *C:\Metreos\PCapService\pcap-service.exe -N*



```
CA\ Command Prompt
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\Documents and Settings\Administrator>cd..
C:\Documents and Settings>cd..
C:\>cd Metreos
C:\Metreos>cd PCapService
C:\Metreos\PCapService>pcap-service.exe -N
15:52:32.506000 CORE: Session 1 read thread starting
1. \Device\NPF_{5A67970E-65B7-4510-9AAA-5701AA27B56D}, Generic dialup adapter
2. \Device\NPF_{5A67970E-65B7-4510-9AAA-5701AA27B56D}, Intel(R) PRO/1000 MT Dual
   Port Network Connection
3. \Device\NPF_{B60817F6-E84C-4662-B07A-C39C938575EA}, Intel(R) PRO/1000 MT Dual
   Port Network Connection
15:52:33.553000 CORE: Session 1 read thread stopping
C:\Metreos\PCapService>_
```

Figure 1: Discovering System Network Adapters

In the sample output shown in Figure 1, above, note that the PC is aware of 3 network adapters. We are not concerned with the Generic dialup adapter. If we were to physically look at the back of the server, the #2 adapter corresponds to the network adapter located on the left, and the #3 adapter corresponds to one on the right.

2. Configure the Network Adapter

With the network adapter you chose above to monitor the span port in mind, open the text file *C:\Metreos\PCapService\pcap-service.config*, and configure *ETHERNET_INTERFACE_ID* to the sequence number shown for the desired network adapter, as illustrated in the example shown in Figure 2.


```
#####
# Config file for pcap-service
#####

# Define ethernet interface id for pcap-service to capture network packets from.
# Run "pcap-service -N" to list available interfaces on running appliance.
ETHERNET_INTERFACE_ID=3

For Help, press F1
```

Figure 2: The PCAP Configuration File

If at any time the configuration is changed, the PCap Service will need to be restarted through the Metreos Web Management Console, mceadmin. The Service Control page found there, as shown in Figure 3, lets you restart this service easily. The service is labeled 'PCap Service'.

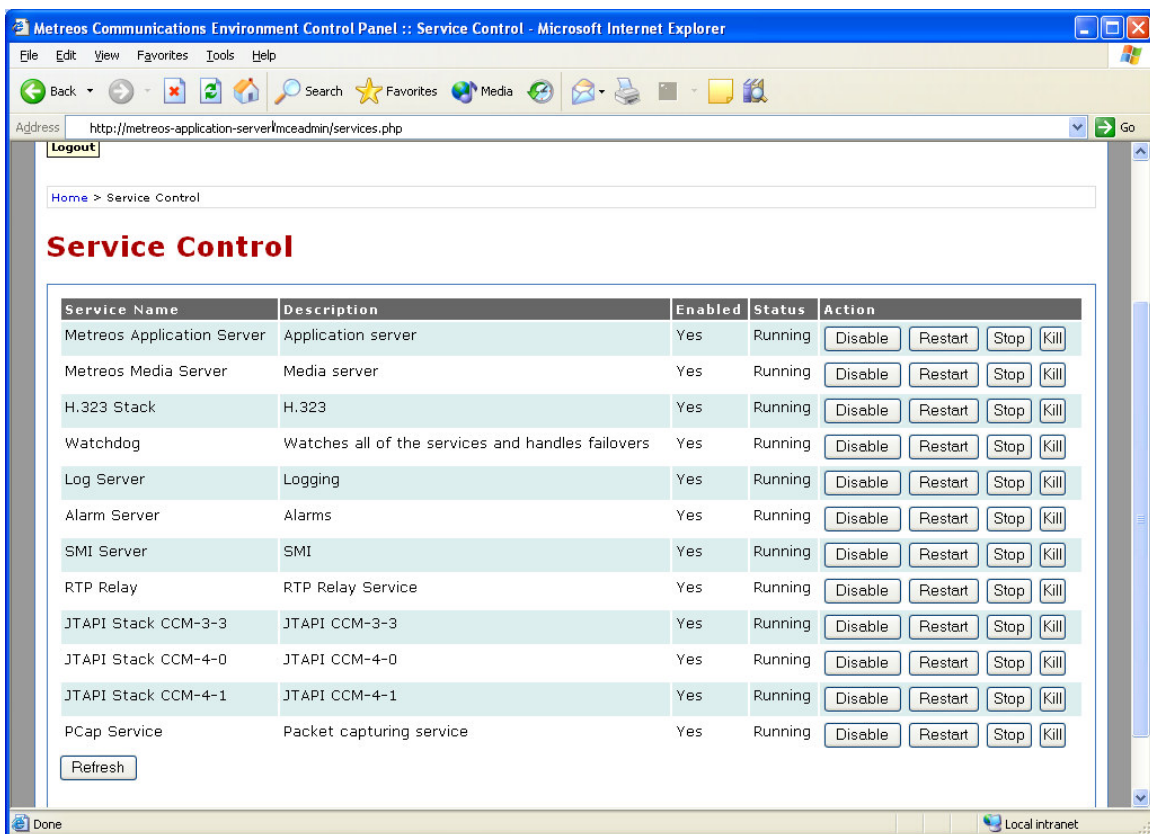


Figure 3: Service Control Panel

CallMonitor Reporting

The Call Monitor application records every instance of usage to a database for the purpose of record keeping. In the case that usage records can not be stored in the database, an XML file is used as a temporary alternative. If at anytime the database becomes available again, the contents of the XML file are written to the database, and the usage redundancy file is then deleted.

A command-line utility is provided in order to retrieve the usage records from the database in a human-readable or comma separated format.

Using the Reporting Tool

Packaged with the Application Server is a simple command line tool which can be used to generate HTML and CSV files.

The tool must be run on the Application Server, and can be found at C:\Metreos\PCapService\CallMonitorReport.exe.

CallMonitorReport.exe Usage

The executable takes up to two arguments:

- -s: Report Start Date
- -e: Report End Date

If both are provided, then the range of time between the date specified by -s and the date specified by -e will be used in querying the database for records.

If only -e is provided, then the reporting tool will report the earliest entry through the date specified by -e.

If only -s is provided, then the reporting tool will report all entries from the date specified by -s through the most current date in the database.

If neither is provided, then the tool will display help information.

The valid formats for specifying a date:

- yyyy-mm-dd
- dd-mon-yy
- mm/dd/yyyy

CallMonitorReport.exe Behavior

Once the date range has been determined, the tool will query the database for all usage records in that range, and generate an HTML and CSV file in the directory C:\monitor_call\AuditLog\.

The files will be of the form [StartDate]_[EndDate].csv and [StartDate]_[EndDate].htm, where [StartDate and EndDate] are of the form yyyy-mm-dd.

Usage Reporting Database Schema

The usage reporting database consists of one table. Each row in the following table corresponds to a field of the usage reporting table.

Database Name: monitor_call
Table Name: monitored_call

Field	Type	Null	Key	Default	Extra
mc_monitored_call_id	Int(10) unsigned		PRI	NULL	Auto Increment
mc_government_agent_number	Varchar(255)				
mc_did_number	Varchar(255)				
mc_insurance_agent_number	Varchar(255)				
mc_customer_number	Varchar(255)				
mc_monitored_sid	Varchar(255)				
mc_start_monitor_timestamp	Timestamp	YES		CURRENT_TIMESTAMP	

Table 1: Table Definition

Field	Description
mc_monitored_call_id	Primary key of the table. Auto incremented upon insertion of a new record.
mc_government_agent_number	The DN of the government agent calling in to monitor a conversation.
mc_did_number	The DID DN used to access the Call Monitor application.
mc_insurance_agent_number	The DN of the insurance agent randomly chosen to be monitored.
mc_customer_number	The DN of the customer that the insurance agent is speaking with.
mc_monitored_sid	The device name, or SID, of the insurance agent's SCCP device.
mc_start_monitor_timestamp	The time that the government agent begins monitoring.

Table 2: Table Description

The system comes preloaded with the database and table already in place. By default, the Call Monitor application writes usage records to this database.

XML Schema

The redundancy usage XML file is a one-to-one mapping of the calls_monitored database table.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="CallRecordTable" xmlns="" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
  <xs:element name="CallRecordTable" msdata:IsDataSet="true">
    <xs:complexType>
      <xs:choice maxOccurs="unbounded">
        <xs:element name="Records">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Record" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:attribute name="governmentAgentNumber" type="xs:string" />
                  <xs:attribute name="did" type="xs:string" />
                  <xs:attribute name="insuranceAgentNumber" type="xs:string" />
                  <xs:attribute name="customerNumber" type="xs:string" />
                  <xs:attribute name="monitoredSid" type="xs:string" />
                  <xs:attribute name="startMonitorTime" type="xs:string" />
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:choice>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Code Listing 1: XML Schema of the application usage redundancy file

```
<?xml version="1.0"?>
<CallRecordTable xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Records>
    <Record governmentAgentNumber="5126872021" did="6213" insuranceAgentNumber="4742"
customerNumber="9892132342" monitoredSid="SEP000011112222" startMonitorTime="2005-09-08
22.27.43:756" />
    <Record governmentAgentNumber="5126872035" did="6217" insuranceAgentNumber="4742"
customerNumber="9892152314" monitoredSid="SEP000011112222" startMonitorTime="2005-09-08
22.28.28:389" />
  </Records>
</CallRecordTable>
```

Code Listing 2: XML example of the application usage redundancy file

The XML redundancy file is written to in the event that the database can not be used to keep usage records. The system will automatically maintain this file. This file can be found at C:\monitor_call\backup\pendingrecords.xml. However, the file will only exist in the case that there is currently an issue in writing records to the database; in the case that the database comes back online, the current contents of the file are inserted into the database, and the file is deleted.

If for some reason neither the file nor the database can be written to, the application will log error messages into the logging system of the Application Server.