



*GLOBAL DATA & MESSAGING*

# **ORBCOMM IP Gateway Developer API Specification**

**IP Gateway  
Version 4.0**

**ORBCOMM, Inc.**

21700 Atlantic Boulevard  
Dulles, Virginia 20166, U.S.A.

Issue Date: July 18, 2006

**G65050710 - Revision C**

**NOTE:** This document contains technical information and descriptions of the ORBCOMM System that reflect the status of the system design and/or planned design as of the date of issue. The contents of this document are subject to change without notice, and no warranty or representation, expressed or implied, is made with respect to its contents.

### **EXPORT CONTROL STATEMENT**

The enclosed technical information is subject to all U.S. export laws and regulations.

The contents of this document, in whole or in part, may only be exported or provided to a non-U.S. citizen in conformance with the U.S. Export Administration Regulations (EAR).

The contents of this document, in whole or in part, may not be diverted, re-exported, transshipped, or otherwise made available to the following groups without an export authorization from the U.S. government:

- A. Any entity or organization proscribed by the U.S. law identified on the Entity List-Supplement No.4 to Part 744 of the EAR, or any person on the Denied Persons List and Specially Designated Nationals-Supplement No. 2 and 3 to Part 764 of the EAR.
- B. Controlled countries: CUBA, IRAQ, IRAN, LIBYA, NORTH KOREA, SERBIA (not including KOSOVO), SUDAN, SYRIA and their nationals.

*Please note that the above information is subject to change and updates will be done as they are received from the Bureau of Export Administration.*

US Export Control Classification Number: 5E991

### **ORBCOMM PROPRIETARY INFORMATION**

This document contains information that is confidential and proprietary to ORBCOMM, Inc., its suppliers, and/or other third parties, and may not be disclosed except under obligations of confidentiality and subject to provisions of a Non-Disclosure Agreement between the proposed recipient and ORBCOMM, Inc., or one of its affiliates.

**NOTE:** ORBCOMM® is a registered service mark of ORBCOMM, Inc. Other marks as used herein are the property of their respective owners.

Copyright © 2005 ORBCOMM, Inc. All Rights Reserved.

## PREFACE

1. This publication describes the specification for API developers.
2. To obtain information relative to this or other ORBCOMM documents, send an Email to:  
`DOCUMENT_CONTROL@ORBCOMM.COM`
3. This document supersedes G65050710 Revision B *Developer API Specification Version 3.1.0*

## TABLE OF CONTENTS

|                    |  |             |
|--------------------|--|-------------|
| <b>Chapter 1</b>   | <b>Introduction .....</b>              | <b>1-1</b>  |
| 1.1                | IP GATEWAY FEATURES OVERVIEW .....     | 1-2         |
| 1.2                | VERSION.....                           | 1-3         |
| 1.3                | ASSUMPTIONS AND RESTRICTIONS.....      | 1-3         |
| 1.4                | TEXT CONVENTIONS .....                 | 1-4         |
| 1.5                | RELATED DOCUMENTS .....                | 1-4         |
| <b>Chapter 2</b>   | <b>IP Gateway Provisioning.....</b>    | <b>2-1</b>  |
| <b>Chapter 3</b>   | <b>Using the IP Gateway API .....</b>  | <b>3-1</b>  |
| 3.1                | INTRODUCTION.....                      | 3-1         |
| 3.2                | REQUEST PARAMETERS .....               | 3-1         |
| 3.3                | RETURN VALUES.....                     | 3-1         |
| 3.3.1              | <i>Result Codes</i> .....              | 3-2         |
| 3.3.2              | <i>Extended Result Codes</i> .....     | 3-2         |
| 3.3.3              | <i>Error Messages</i> .....            | 3-2         |
| <b>CHAPTER 4</b>   | <b>AUTHENTICATE .....</b>              | <b>4-1</b>  |
| <b>CHAPTER 5</b>   | <b>SENDMESSAGE .....</b>               | <b>5-1</b>  |
| <b>CHAPTER 6</b>   | <b>QUERYMESSAGESTATUS.....</b>         | <b>6-1</b>  |
| <b>Chapter 7</b>   | <b>DeleteMessage.....</b>              | <b>7-1</b>  |
| <b>Chapter 8</b>   | <b>QueryDeviceStatus.....</b>          | <b>8-1</b>  |
| <b>CHAPTER 9</b>   | <b>RETRIEVEMESSAGES.....</b>           | <b>9-1</b>  |
| <b>CHAPTER 10</b>  | <b>SETMESSAGEFLAG .....</b>            | <b>10-1</b> |
| <b>Chapter 11</b>  | <b>HTTP Notification .....</b>         | <b>11-1</b> |
| <b>CHAPTER 12</b>  | <b>REFRESH .....</b>                   | <b>12-1</b> |
| <b>CHAPTER 13</b>  | <b>LOGOUT .....</b>                    | <b>13-1</b> |
| <b>Appendix A–</b> | <b>HTTP URL Encoding Scheme .....</b>  | <b>A-1</b>  |
| <b>Appendix B–</b> | <b>XML Response Format .....</b>       | <b>B-1</b>  |
| <b>Appendix C–</b> | <b>Frequently Asked Questions.....</b> | <b>C-1</b>  |

## LIST OF TABLES

|  |      |
|--|------|
| TABLE 1-1 -- POTENTIAL IP GATEWAY APPLICATIONS .....               | 1-2  |
| TABLE 4-1 -- AUTHENTICATE PARAMETERS .....                         | 4-2  |
| TABLE 4-2 -- AUTHENTICATE RETURN VALUES.....                       | 4-2  |
| TABLE 4-3 -- RESULT CODES .....                                    | 4-2  |
| TABLE 4-4 -- EXTENDED RESULT CODES .....                           | 4-3  |
| TABLE 5-1 -- SENDMESSAGE PARAMETERS .....                          | 5-1  |
| TABLE 5-2 -- FORMAT OF <i>SEND_TIME</i> PARAMETER.....             | 5-2  |
| TABLE 5-3 -- RETURN VALUES .....                                   | 5-2  |
| TABLE 5-4 -- RESULT CODES .....                                    | 5-3  |
| TABLE 5-5 -- EXTENDED RESULT CODES .....                           | 5-3  |
| TABLE 6-1 -- QUERYMESSAGESTATUS PARAMETERS .....                   | 6-2  |
| TABLE 6-2 -- QUERYMESSAGESTATUS RETURN VALUES .....                | 6-2  |
| TABLE 6-3 -- QUERYMESSAGESTATUS RESULT CODES .....                 | 6-3  |
| TABLE 6-4 -- QUERYMESSAGESTATUS MESSAGE STATUS CODES .....         | 6-3  |
| TABLE 6-5 -- QUERYMESSAGESTATUS EXTENDED RESULT CODES.....         | 6-3  |
| TABLE 6-6 -- QUERYMESSAGESTATUS STATE CODES.....                   | 6-4  |
| TABLE 6-7 -- QUERY MESSAGE STATUS STATUS CODES.....                | 6-4  |
| TABLE 6-8 -- QUERYMESSAGESTATUS DIAGNOSTIC CODES.....              | 6-5  |
| TABLE 7-1 -- REQUIRED PARAMETERS .....                             | 7-2  |
| TABLE 7-2 -- RETURN VALUES .....                                   | 7-2  |
| TABLE 7-3 -- RESULT CODES .....                                    | 7-2  |
| TABLE 7-4 -- EXTENDED RESULT CODES .....                           | 7-3  |
| TABLE 8-1 -- REQUEST PARAMETERS.....                               | 8-2  |
| TABLE 8-2 -- QUERYDEVICESTATUS RETURN VALUES.....                  | 8-2  |
| TABLE 8-3 -- QUERYDEVICESTATUS RESULT CODES.....                   | 8-3  |
| TABLE 8-4 -- QUERYDEVICESTATUS EXTENDED RESULT CODES .....         | 8-3  |
| TABLE 9-1 -- EFFECT OF MESSAGE_ID ON RETRIEVEMESSAGES RESULTS..... | 9-3  |
| TABLE 9-2 -- RETRIEVEMESSAGES PARAMETERS .....                     | 9-4  |
| TABLE 9-3 -- RETRIEVEMESSAGES RETURN VALUES .....                  | 9-5  |
| TABLE 9-4 -- RETRIEVEMESSAGES MESSAGE STATUS CODES.....            | 9-6  |
| TABLE 9-5 -- RETRIEVEMESSAGES RESULT CODES .....                   | 9-6  |
| TABLE 9-6 -- RETRIEVEMESSAGES EXTENDED RESULT CODES .....          | 9-6  |
| TABLE 10-1 -- SETMESSAGEFLAG REQUIRED PARAMETERS.....              | 10-1 |
| TABLE 10-2 -- SETMESSAGEFLAG RETURN VALUES .....                   | 10-1 |
| TABLE 10-3 -- SETMESSAGEFLAG RESULT CODES.....                     | 10-2 |
| TABLE 10-4 -- SETMESSAGEFLAG EXTENDED RESULT CODES .....           | 10-2 |
| TABLE 12-1 -- REFRESH PARAMETER .....                              | 12-1 |
| TABLE 12-2 -- REFRESH RETURN VALUES .....                          | 12-1 |
| TABLE 12-3 -- REFRESH RESULT CODES .....                           | 12-2 |
| TABLE 13-1 -- LOGOUT PARAMETER .....                               | 13-1 |
| TABLE 13-2 -- LOGOUT RETURN VALUE .....                            | 13-1 |
| TABLE 13-3 -- LOGOUT RESULT CODES .....                            | 13-1 |

# Chapter 1 Introduction

ORBCOMM is the world's first commercial provider of global low-Earth orbit satellite data and messaging services. The ORBCOMM system enables businesses to track remote and mobile assets such as trailers, railcars and heavy equipment; monitor remote utility meters and oil and gas storage tanks, wells and pipelines; and stay in touch with remote workers anywhere on the globe.

This document introduces ORBCOMM IP Gateway. The IP Gateway interface employs standard HTTP requests and XML responses over the Internet.. The IP Gateway resides at ORBCOMM and contains proprietary software to process and route messages, status requests and commands to and from the designated remote devices.

The IP Gateway provides proven wireless data communication solutions that can increase a customer's competitive advantage by providing global coverage. ORBCOMM IP Gateway solution can also integrate with a customer's existing legacy systems.

ORBCOMM and its business partners provide powerful, end-to-end data and messaging communications solutions to a wide variety of industries. Some of the applications that could utilize the IP Gateway are listed below.

- |                         |  |
|-------------------------|--|
| <b>Energy:</b>          | ORBCOMM offers state-of-the-art energy monitoring solutions for everything from oil and gas storage tanks, wells and pipelines to remote hydrological stations.  |
| <b>Utilities:</b>       | ORBCOMM's cost-effective, two-way data and messaging solutions perform many utility services, such as meter reading, service connects/disconnects, power outage detection, as well as water quality monitoring and energy and natural gas use.   |
| <b>Transportation:</b>  | ORBCOMM provides targeted solutions for a variety of transportation industries including trailer tracking and monitoring for the trucking and shipping industry, rail car and cargo monitoring and in-flight weather information and messaging for small aircraft. ORBCOMM's complete end-to-end transportation solutions can help companies manage their assets anytime anywhere. |
| <b>Heavy Equipment:</b> | Track and monitor your heavy equipment in the field anywhere on the globe. ORBCOMM solutions for the construction industry can help you track usage data, asset location, possible theft, engine hours and other critical machine operating parameters. ORBCOMM helps you keep your fleet up and running at the least possible cost.   |

|                                    |  |
|------------------------------------|--|
| <b>Government:</b>                 | ORBCOMM is working with a variety of government customers to provide innovative communications solutions such as advanced weather sensing and surveillance programs.   |
| <b>Environmental and Weather:</b>  | ORBCOMM has teamed up with several companies to provide cost-efficient, remote environmental monitoring and weather mapping services.  |
| <b>Chemical and Petrochemical:</b> | With one seamless communications system, you can monitor and manage your complete inventory of stored and transported chemicals.   |
| <b>Marine:</b>                     | There can be huge losses when perishable goods are destroyed, when cargo is lost or stolen or when shipments fail to reach their destination on time. With ORBCOMM's tracking and monitoring solutions for the marine industry, you can know the location and arrival time of your fishing vessels or barges. You also can be alerted if a shipment has been moved outside the normal transportation route, if a container has been opened or if its contents have spoiled while in route. |
| <b>Messaging:</b>                  | Stay connected and in touch with ORBCOMM's global two-way data messaging solutions while at work or at play, on land or at sea. The ORBCOMM messaging system makes it possible to send and receive brief messages, check in with the office or even call for help if necessary-even in the most remote locations.  |
| <b>Automotive:</b>                 | ORBCOMM is developing reliable, cost-effective communications solutions for the automotive industry. ORBCOMM automotive applications are proposed to support a variety of applications ranging from emergency and roadside assistance to stolen vehicle tracking and fleet management.   |

**Table 1-1 -- Potential IP Gateway applications**

## 1.1 IP Gateway Features Overview

The IP Gateway provides the following features:

- HTTP/XML Host Interface. This is an interconnection option to the ORBCOMM network that requires no specialized interface software and could be used by a customer with a browser such as Internet Explorer. Software modules and APIs are available to ease application development.

- Message Status. The customer application can request the status of messages sent to ORBCOMM subscriber communicators (SCs) and the time of the last ORBCOMM radio communication from the SC.
- Device Status. The customer application can query the SC status.
- Scheduled Message Delivery. SC-terminated messages can be submitted for delivery at a specified date and time. This feature could work in conjunction with an SC waking up from a power-saving sleep mode.
- HTTP Notification service. Provides the capability of sending an HTTP notification to a specified URL when an SC-originated message is received at the IP Gateway for any SC associated with a customer profile.
- Security. The Gateway provides a secure Internet interface through the use of a username/password and addressing restrictions to designated groups. Another option available for added security is through the use of Secure Socket Layer (SSL) encryption on the transfer of data.

## 1.2 Version

This document describes IP Gateway version 4.0. IP Gateway 4.0 provides support for previous versions of the IP Gateway, as well as providing some new features and correcting some inconsistencies in operation.

To accommodate the changes while maintaining backward compatibility, the API has been assigned a version number as well. Version 1 of the IP Gateway API was the only version supported by all versions of the IP Gateway prior to 4.0, and is also supported by version 4.0

Version 2 of the API is released with, and supported by IP Gateway version 4.0. It can be accessed as described herein, with one additional command and extensions to several existing commands. Where Version 2 enhances existing commands, the new feature set is accessed via the VERSION=2 parameter-value pair in the command URL. This parameter value pair can be added to any command URL regardless of whether any new features are offered. For all API version 1 commands, use of version 1 is assumed unless version 2 is specified.

## 1.3 Assumptions and Restrictions

In order to invoke an API the user must POST an HTTP1.1 request to the URL of the ORBCOMM IP Gateway. Every request must follow the rules and conventions of HTTP URL syntax (refer to *Appendix A*). Note that HTTP URLs are case sensitive. If URLs are not properly encoded, the API makes a best effort to inform the invoker of the error. However, due to the nature of the protocol, this is not always possible.

The responses from the IP Gateway are in XML format and must be parsed by an XML parser. The XML response tags are not to be assumed to be in a certain order.

A user must not attempt a connection to the IP Gateway at a frequency greater than once per minute.



## 1.4 Text Conventions

This document uses the following text conventions:

Text in a `fixed-width font` indicates information passed to the IP Gateway in a URI, or XML received from the IP Gateway.

IP Gateway commands except where specified as a part of a URI, are identified with a SMALL CAPS font.

Parameters for IP Gateway commands, except where used as part of a URI, are identified by *italics*.

## 1.5 Related Documents

- ORBCOMM IP Gateway (ORBCOMM document no. G65RT0711)
- “Hypertext Transfer Protocol -- HTTP/1.1”. RFC 2068, R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee. January 1997.
- Customer Service Provisioning Guide IP Gateway Orbcomm Document number D21TD0741
- ORBCOMM Provisioning Processes – April 2000

## Chapter 2 IP Gateway Provisioning

Each user (Account Name or company) must pre-register with ORBCOMM to obtain an IP Gateway account name and password in order to be able to access the IP Gateway. Figure 2-1, “Customer Service IP Gateway Account Set-up Form”, shows the details regarding the information required to create an IP Gateway account.

After an account is created, an SC that will use the IP Gateway must also be provisioned to do so. In addition to normal SC provisioning information, the IP Gateway account to which the SC is to be provisioned must be specified. Additionally, speed dial #1, will be set to the address of the IP Gateway.

Once an account is established, and one or more SCs are associated with it, the SCs can be accessed using this API. Each SC on an account is available to anyone using the account, so customers may want to establish different accounts for different groups of SCs.

Figure 2-1 -- Customer Service IP Gateway Account Set-up Form



---

***IP Gateway Account Set-up Form***

**Account Name (Company):**

---

**Requestor's Name:**

**Phone Number:**

**Date of Activity:**

---

**IP Gateway Account Information: (All Fields Required)**

**New Account:** ☐ **Change to Account Information:** ☐

**Subscriber Group ID:**

**Password:**

---

**Instructions:**

**Account Name (Company):** Provide previously established Account Name. *Provide company name or individual name, as documented in the signed ORBCOMM contract using upper and lower case as appropriate.*

**Requestor's Name:** Provide your complete name.

**Phone Number:** Provide your phone number.

**Date of Activity:** Enter the date on which activation is to become effective.

**Indicate** if the account is New or if you are making a change to an existing account by checking the appropriate box.

**Subscriber Group ID:** Select your desired Subscriber Group ID.

**Password:** Select your desired password.

***Please return completed form to ORBCOMM Customer Service at  
Customer\_service@orbcomm.com***

## Chapter 3 Using the IP Gateway API

### 3.1 Introduction

This API is arranged into several sections. Each section contains the formats for a particular command and the associated responses. An example for each API HTTP command and its XML result is provided. The appendices contain additional information on URL encoding, XML parsing, and frequently asked questions and answers.

A typical application would perform the following pseudo logic to send messages, query status, retrieve messages, etc:

1. Send **AUTHENTICATE** HTTP command.
2. Wait for a response from the IP Gateway with a session ID number.
3. Send **SENDMESSAGE** command.
4. Wait for a response from IP Gateway with a confirmation number.
5. Send a **QUERYMESSAGESTATUS** command to check the message delivery status.
6. Wait for a response from the IP Gateway with result information.
7. Send a **RETRIEVEMESSAGES** command.
8. Wait for a response from the IP Gateway with a list of ORBCOMM SC-originated messages.
9. Send a **REFRESH** command to preserve the session.
10. Send a **LOGOUT** command to close the session.

### 3.2 Request Parameters

Each of the API commands has a request in the form of an HTTP POST. The HTTP POST parameter values must conform to the HTTP specification and must be URL encoded. The POST parameters for which there are default values do not need to be a part of the request, and, if excluded, the default values will be used. For example, with the **SENDMESSAGE** API request, you must send a *SESSION\_ID* and a *DEVICE\_ID*, however, all the other parameters (*SEND\_TIME*, *MESSAGE\_SUBJECT*, etc.) are optional.

### 3.3 Return Values

After an API HTTP command has been issued, the response comes back as an XML document. The return values for some commands will contain information that needs to be parsed for subsequent calls to other API commands. For example, with **AUTHENTICATE**, the return values include a *SESSION\_ID* tag, which will contain the session identifier for all subsequent API HTTP commands.

### **3.3.1 Result Codes**

The result codes identified by the RESULT tags are used as an indicator for the application as to the state of the request. If the result code value is -1, it indicates that there was an invalid session. For each API command, the additional possible result codes will vary.

### **3.3.2 Extended Result Codes**

The extended result codes identified by the EXTEND\_DESC tags are to help troubleshoot error conditions. They are not to be used for case logic. The result codes should be used for logic of success/failure.

### **3.3.3 Error Messages**

A Java exception error 404 occurs when the Gateway cannot distinguish a command or request.

## Chapter 4 AUTHENTICATE

This function is used to validate the user via password authentication. The function returns a unique *SESSION\_ID* that must be used for all subsequent calls to the ORBCOMM IP Gateway. Once the user receives a valid session ID, the user may have access to devices that are associated with that account. The session ID will be valid for a configured period of time of inactivity; after that period expires the user must re-authenticate. The inactivity period is a system parameter and will be configured between 10 to 60 minutes.

### Example 1 (without *VERSION* parameter):

`http://host/Authenticate?LOGIN=username&PSSWD=password`

where host is the IP address of the IP Gateway.

### XML result:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <AUTHENTICATE>
  <SESSION_ID>2191112113663751</SESSION_ID>
  <RESULT>1</RESULT>
  <EXTEND_DESC>Success</EXTEND_DESC>
</AUTHENTICATE>
```

The **Authenticate** function has been enhanced in IP Gateway Version 4.0 to include the *VERSION* parameter. When *VERSION*=2 is specified in the request, additional information is returned in the response. When *VERSION*=1 or none is specified in the request, the response is compatible to IP Gateway Version 3.

### Example 2 (with *VERSION* parameter specified):

`http://host/Authenticate?LOGIN=username&PSSWD=password&VERSION=2`

### XML result:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <AUTHENTICATE>
  <SESSION_ID>2211112113833898</SESSION_ID>
  <RESULT>1</RESULT>
  <EXTEND_DESC>Success</EXTEND_DESC>
  <SESS_EXPIRE_TIME>2005/03/29 17:00:33</SESS_EXPIRE_TIME>
  <TOTAL_ACTIVE_SESS>7</TOTAL_ACTIVE_SESS>
</AUTHENTICATE>
```

If a *SESSION\_ID* of -1 is returned, the Authenticate has failed and the application must re-authenticate.

| Name           | Type    | Max Size | Default | Description   |
|----------------|---------|----------|---------|---|
| <i>LOGIN</i>   | String  | 128      |         | User name of profile  |
| <i>PSSWD</i>   | String  | 16       |         | The password assigned to the given profile  |
| <i>VERSION</i> | Numeric | 1        | 1       | 1=backward compatible to IP Gateway Version 3.x; 2=return additional information  |
| <i>URL</i>     | String  | 1024     | Null    | Fully qualified URI used for HTTP notification of mobile originated messages. This is only required if the host application needs SC-originated message notification. See Chapter 11 for more information |

**Table 4-1 -- AUTHENTICATE Parameters**

| Name              | Type    | Max Size | Description   |
|-------------------|---------|----------|---|
| SESSION_ID        | Numeric | 32       | Session identifier that must be used in subsequent operations             |
| RESULT            | String  | 2        | Contains error code if operation failed                                   |
| EXTEND_DESC       | String  | 1024     | Verbose error description   |
| SESS_EXPIRE_TIME  | String  | 19       | If <i>VERSION</i> =2, returns date/time of when this session expires      |
| TOTAL_ACTIVE_SESS | Numeric | 3        | If <i>VERSION</i> =2, returns total active sessions for the login account |

**Table 4-2 -- AUTHENTICATE Return Values**

| RESULT | Description            |
|--------|------------------------|
| 1      | Success                |
| 10     | PARAM_FORMAT_ERROR     |
| 20     | FORM_VARIABLE_MISSING  |
| 30     | DB_ERROR               |
| 40     | DB_LOOKUP_FAILED       |
| 50     | INTERNAL_SERVLET_ERROR |

**Table 4-3 -- Result Codes**

| EXTEND_DESC  |
|--|
| Success  |
| Invalid Login/Password                                     |
| The required parameter "LOGIN" was not present in the form |
| The required parameter "PSSWD" was not present in the form |
| Login and/or Password contained bad formats                |

**Table 4-4 -- Extended Result Codes**



## Chapter 5 SENDMESSAGE

The SENDMESSAGE command allows the user to send ASCII or Binary messages from the IP Gateway to a SC.

The AUTHENTICATE command must be called before the SENDMESSAGE command to obtain the required *SESSION\_ID* value. If the response to SENDMESSAGE is “-1, Session Invalid”, then the application must re-authenticate as the *SESSION\_ID* has expired or has been invalidated.

### Example:

```
http://host/SendMessage?SESSION_ID=2211112113833898&DEVICE_ID=sc1&NETWORK_ID=3&MESSAGE_SUBJECT=testmsg1&MESSAGE_BODY_TYPE=0&MESSAGE_BODY=test msg body&MESSAGE_PRIORITY=0&SEND_TIME=*****
```

### XML results:

```
<?xml version="1.0" ?>
<SENDMESSAGE>
  <CONF_NUM>100000065</CONF_NUM>
  <RESULT>1</RESULT>
  <EXTEND_DESC>Success</EXTEND_DESC>
</SENDMESSAGE>
```

| Name                     | Type    | Max Size | Default Value | Description  |
|--------------------------|---------|----------|---------------|--|
| <i>SESSION_ID</i>        | Numeric | 32       |               | Session Identifier obtained via Authenticate                             |
| <i>DEVICE_ID</i>         | String  | 128      |               | The device alias for the destination SC                                  |
| <i>NETWORK_ID</i>        | Numeric | 2        | 3             | 3 = ORBCOMM  |
| <i>MESSAGE_SUBJECT</i>   | String  | 128      | ""            | The subject of the message. The default is "" (i.e., a blank subject)    |
| <i>MESSAGE_BODY_TYPE</i> | Numeric | 1        | 0             | 0 = ASCII<br>1 = Binary ( <i>MESSAGE_BODY</i> must be Base64 encoded)    |
| <i>MESSAGE_BODY</i>      | String  | 2000     | ""            | The body of the message. The default is "" (i.e., a blank message body). |
| <i>MESSAGE_PRIORITY</i>  | Numeric | 1        | 1             | 0 = Non-urgent<br>1 = Normal<br>2 = Urgent                               |
| <i>SEND_TIME</i>         | String  | 12       | *****         | Time to send message. See "Scheduled Messages" below.                    |

Table 5-1 -- SENDMESSAGE Parameters

Note that the lengths of *MESSAGE\_SUBJECT* and *MESSAGE\_BODY* are limited. If the submitted size of either exceeds the maximum value, the message will be truncated and the *EXTEND\_DESC* will contain a warning that truncation occurred.

For binary messages, the message must be base64 encoded. The resulting base64 encoding may contain unsafe characters for the HTTP URL POST command; therefore, the base64-encoded message must also be URL encoded.

### Scheduled Messages

The *SENDMESSAGE* command contains a *SEND\_TIME* field that is used to schedule a message to be sent at a predetermined time. The *SEND\_TIME* parameter should be used with the format of “YYYYMMDDhhmm” as illustrated in Table 5-2.

| Symbol | Meaning            | Presentation | Example |
|--------|--------------------|--------------|---------|
| Y      | year               | (Number)     | 2000    |
| M      | month in year      | (Number)     | 03      |
| D      | day in month       | (Number)     | 24      |
| h      | hour in day (0~23) | (Number)     | 11      |
| m      | minute in hour     | (Number)     | 30      |

**Table 5-2 -- Format of *SEND\_TIME* parameter**

The *SEND\_TIME* format is parsed right to left. The “\*” character is used as a wildcard: the first \* encountered will mark the rest of the format to wild cards. The only valid formats follow:

January 31, 2000, 6:00pm GMT: 200001311800

January 31, current year, 6:00pm GMT: \*\*\*\*01311800

Current month 31, current year, 6:00pm GMT: \*\*\*\*\*311800

Current month current day, current year, 6:00pm GMT: \*\*\*\*\*1800

If the user specifies a date/time combination that is earlier than the current date/time, the message will be sent immediately.

| Name        | Type    | Max Size | Description                             |
|-------------|---------|----------|---|
| CONF_NUM    | Numeric | 32       | Confirmation number of message          |
| RESULT      | Numeric | 2        | Contains error code if operation failed |
| EXTEND_DESC | String  | 1024     | Verbose error description               |

**Table 5-3 -- Return Values**

| RESULT | Description            |
|--------|------------------------|
| -1     | Invalid Session ID     |
| 1      | Success                |
| 10     | PARAM_FORMAT_ERROR     |
| 20     | FORM_VARIABLE_MISSING  |
| 30     | DB_ERROR               |
| 40     | DB_LOOKUP_FAILED       |
| 50     | INTERNAL_SERVLET_ERROR |

**Table 5-4 -- Result Codes**

| EXTEND_DESC   |
|---|
| Success   |
| Session Invalid   |
| The required parameter "SESSION_ID" was not present in the form |
| The required parameter "DEVICE_ID" was not present in the form  |
| Illegal value (n) for NETWORK_ID                                |
| Illegal value (n) for MESSAGE_BODY_TYPE                         |
| Invalid Device ID   |
| Body of Message Truncated                                       |
| Subject Truncated   |
| Illegal length (n) for SEND_TIME                                |
| Illegal value in SEND_TIME. Letter is not permitted.            |

**Table 5-5 -- Extended Result Codes**

## Chapter 6 QUERYMESSAGESTATUS

This function is used to query status of a specific message. After a SENDMESSAGE request has been delivered to the IP Gateway, the system will return a CONF\_NUM as a reference to that SC-terminated message. When the user wishes to check the message delivery status and determine if the message has been delivered to the SC, the user sends that number as the *CONF\_NUM* parameter in the QUERYMESSAGESTATUS request.

AUTHENTICATE must be called before QUERYMESSAGESTATUS to obtain the required *SESSION\_ID* value. If the response to QUERYMESSAGESTATUS is “-1, Session Invalid,” then the application must re-authenticate as the *SESSION\_ID* has expired or has been invalidated.

### Example 1:

```
http://host/QueryMessageStatus?SESSION_ID=1231112116098070&CONF_NUM=10000065&MESSAGE=1
```

### XML results (without *VERSION* parameter):

```
<?xml version="1.0" encoding="UTF-8" ?>
- <QUERYMESSAGESTATUS>
  <MESSAGE_TIME>2005/03/29 11:59:32</MESSAGE_TIME>
  <RESULT>1</RESULT>
  <MESSAGE_FROM>test1</MESSAGE_FROM>
  <MESSAGE_TO>sc1</MESSAGE_TO>
  <MESSAGE_SUBJECT>testmsg1</MESSAGE_SUBJECT>
  <MESSAGE_ENCODING>ASCII</MESSAGE_ENCODING>
  <MESSAGE_BODY>test msg body</MESSAGE_BODY>
  <MESSAGE_STATUS>Delivered</MESSAGE_STATUS>
  <EXTEND_DESC>Success</EXTEND_DESC>
</QUERYMESSAGESTATUS>
```

The QUERYMESSAGESTATUS function has been enhanced in IP Gateway Version 4.0 to include the *VERSION* parameter. When *VERSION*=2 is specified in the request, additional information is returned in the response. When *VERSION*=1 or none is specified in the request, the response is compatible to IP Gateway Version 3.

### Example 2 (with *VERSION* parameter specified):

```
http://host/QueryMessageStatus?SESSION_ID=1231112116098070&CONF_NUM=10000065&MESSAGE=1&VERSION=2
```

### XML results:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <QUERYMESSAGESTATUS>
  <MESSAGE_TIME>2005/03/29 11:59:32</MESSAGE_TIME>
  <RESULT>1</RESULT>
  <MESSAGE_FROM>test1</MESSAGE_FROM>
  <MESSAGE_TO>sc1</MESSAGE_TO>
  <MESSAGE_PRIORITY>normal</MESSAGE_PRIORITY>
  <MESSAGE_SUBJECT>testmsg1</MESSAGE_SUBJECT>
```

```
<MESSAGE_ENCODING>ASCII</MESSAGE_ENCODING>
<MESSAGE_BODY>test msg body</MESSAGE_BODY>
<MESSAGE_STATUS>Delivered</MESSAGE_STATUS>
<EXTEND_DESC>Success</EXTEND_DESC>
</QUERYMESSAGESTATUS>
```

STATE, STATUS and DIAG return values are only available for ORBCOMM SC-terminated messages. They describe the status of the message within the OMS and are informational only.

| Name       | Type    | Max Size | Default | Description   |
|------------|---------|----------|---------|---|
| SESSION_ID | Numeric | 32       |         | Session identifier obtained via AUTHENTICATE  |
| CONF_NUM   | Numeric | 32       |         | Confirmation number of message  |
| MTAG       | Number  | 1        | 0       | 0 = QUERYMESSAGESTATUS is XML parent<br>1 = Message is XML parent                   |
| MESSAGE    | Numeric | 1        | 0       | 0 = Do not return message body<br>1 = Return message body                           |
| VERSION    | Numeric | 1        | 1       | 1=backward compatible to IP Gateway<br>Version 3<br>2=return additional information |

Table 6-1 -- QUERYMESSAGESTATUS Parameters

| Name             | Type    | Max Size | Description   |
|------------------|---------|----------|---|
| NETWORK_ID       | Numeric | 2        | 3 = ORBCOMM   |
| MESSAGE_TIME     | String  | 19       | Date/Time message was originated/terminated   |
| RESULT           | Numeric | 2        | Contains error code if operation failed   |
| MESSAGE_FROM     | String  | 128      | Originator Address  |
| MESSAGE_TO       | String  | 128      | Destination Address   |
| MESSAGE_PRIORITY | String  | 10       | IF VERSION=2, priority of message is returned with one of the following: non-urgent; normal; urgent |
| MESSAGE_SUBJECT  | String  | 128      | Subject of the message  |
| MESSAGE_ENCODING | String  | 6        | Body encoding type (ASCII/Binary)   |
| MESSAGE_BODY     | String  | 2000     | Body of the message   |
| MESSAGE_STATUS   | String  | 16       | Status code from IP Gateway   |
| EXTEND_DESC      | String  | 1024     | Description of result codes   |
| STATE            | String  | 1024     | State codes from OMS  |
| STATUS           | String  | 1024     | Status codes from OMS   |
| DIAG             | String  | 1024     | Diagnostic codes from OMS   |

Table 6-2 -- QUERYMESSAGESTATUS Return Values

| RESULT | Description   |
|--------|---|
| -1     | Invalid Session ID  |
| 1      | Success; record found but not able to communicate with OMS; record exists in IP Gateway but not at OMS. |
| 10     | PARAM_FORMAT_ERROR  |
| 20     | FORM_VARIABLE_MISSING   |
| 30     | DB_ERROR  |
| 40     | DB_LOOKUP_FAILED  |
| 50     | INTERNAL_SERVLET_ERROR  |

Table 6-3 -- QUERYMESSAGESTATUS Result Codes

| MESSAGE_STATUS | Description   |
|----------------|---|
| Pending        | The message has been accepted by the IP Gateway for delivery  |
| InTransit      | The message has been submitted to OMS for delivery. If VERSION=1, the message has or has not been received by the OMS. If VERSION=2, the message has not been received by the OMS. It is recommended that VERSION=2 be used in the request for true status. |
| Delayed        | The message is being held by the IP Gateway for delivery until the specified time that was requested in the delivery send_time option   |
| ReceivedOMS    | If VERSION=2, this indicates the message was received by the OMS.   |
| Delivered      | The message was delivered and an ACK has been received from the OMS   |
| Exhausted      | If VERSION=2, this indicates the OMS has exhausted all attempts to deliver the message to the SC.   |
| Undelivered    | The message has been rejected by the OMS  |
| SC Originated  | The message arrived to the IP Gateway via an SC   |

Table 6-4 -- QUERYMESSAGESTATUS Message Status codes

| EXTEND_DESC  |
|--|
| Success  |
| Session Invalid  |
| The required parameter "SESSION_ID" was not present in the form.                             |
| The required parameter "CONF_NUM" was not present in the form.                               |
| Invalid Confirmation Number  |
| IMD Query has Failed ( <b>VERSION=1 only</b> )   |
| Unable to communicate with OMS on GCC(n) ( <b>VERSION=2 only, where n indicates GCC ID</b> ) |
| Record not found in OMS DB at GCC(n) ( <b>VERSION=2 only</b> )                               |

Table 6-5 -- QUERYMESSAGESTATUS Extended Result Codes

| STATE   |
|---|
| Subscriber Terminated Message Received by OMS |
| Subscriber profile lookup successful          |
| Subscriber profile lookup failure             |
| Message is stored awaiting delivery           |
| Message cannot be stored                      |
| Satellite available                           |
| Satellite not available                       |
| System resource available                     |
| System resource not available                 |
| Outbound message session successful           |
| Outbound message session failure              |
| Message queue update successful               |
| Message queue update failure                  |

**Table 6-6 -- QUERYMESSAGESTATUS State Codes**

| STATUS                                |
|---------------------------------------|
| Message transmission failure          |
| Unable to transmit message            |
| Message No Conversion                 |
| Reserved Future Use                   |
| Message inquiry response              |
| Status unknown                        |
| Message delivered and waiting for ACK |
| User Abort                            |
| Message sent OK                       |

**Table 6-7 -- Query Message Status Status Codes**

| DIAG                                    |
|---|
| Originator/Recipient name unrecognized  |
| Originator/Recipient name ambiguous     |
| MTA congestion                          |
| Loop detected                           |
| Recipient unavailable                   |
| Transmission time out                   |
| Body part type supported                |
| Content too long                        |
| Convert impractical                     |
| Conversion prohibited                   |
| Conversion not registered               |
| Invalid parameter                       |
| Reserved for future use                 |
| SC is not registered                    |
| Invalid PIN code                        |
| NCC not found                           |
| Insufficient priority                   |
| No response from satellite              |
| SC access restriction                   |
| SC registration expired                 |
| Inbound message already exists          |
| Inbound message number error            |
| File error                              |
| Database error                          |
| No additional info                      |
| Max retry exhausted for inbound message |
| GlobalGgram not permitted               |
| Satellite not in view                   |
| POS_REP not available                   |
| NO_POS detected                         |
| Exceed GlobalGram size                  |
| HCS has no message                      |
| HCS has no small message                |
| Satellite prohibit GlobalGram           |
| Requested message deleted               |
| No_Stored_sats                          |
| Preempted                               |
| Queue failure                           |
| Exceed inbound queue size               |
| Illegal ORBCOMM O/R                     |

**Table 6-8 -- QUERYMESSAGESTATUS Diagnostic Codes**



## Chapter 7 DeleteMessage

This function is used to delete a specific message. After a SENDMESSAGE request has been delivered to the IP Gateway, the system will return a CONF\_NUM as a reference to that SC-terminated message. If the user wishes to delete the message, the user sends that number as the *CONF\_NUM* parameter in the DELETMESSAGE request.

DELETMESSAGE is only valid if the message is not actively being sent by the OMS to the SC. This function will delete SC-terminated messages that are in certain states:

- in the in the IP Gateway pending delivery to the OMS,
- at the OMS but not actively being sent by the OMS to the SC, or
- already marked delivered or undeliverable to the SC.

If the message specified for deletion is at the OMS but is actively being sent, or cannot be deleted for some other reason, an error will be returned.

AUTHENTICATE must be called before DELETMESSAGE to obtain the required *SESSION\_ID* value. If the response to DELETMESSAGE is “-1, Session Invalid,” then the application must re-authenticate as the *SESSION\_ID* has expired or has been invalidated.

### Example 1:

```
http://host/DeleteMessage?SESSION_ID=2231112123701635&CONF_NUM=100000073
```

### XML results:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <DELETMESSAGE>
  <RESULT>1</RESULT>
  <EXTEND_DESC>Success</EXTEND_DESC>
</DELETMESSAGE>
```

The DELETMESSAGE function has been enhanced in IP Gateway Version 4.0 to include the *VERSION* parameter. When *VERSION*=2 is specified in the request, additional information can be returned in the response, depending on the result of the request. When *VERSION*=1 or none is specified, the response is compatible to IP Gateway Version 3.

### Example 2 (with VERSION parameter specified):

```
http://host/DeleteMessage?SESSION_ID=2231112123701635&CONF_NUM=100000073&VERSION=2
```

### XML results:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <DELETMESSAGE>
  <RESULT>1</RESULT>
  <EXTEND_DESC>Success</EXTEND_DESC>
</DELETMESSAGE>
```

| Name              | Type    | Max Size | Default | Description   |
|-------------------|---------|----------|---------|---|
| <i>SESSION_ID</i> | Numeric | 32       |         | Session identifier obtained via Authenticate                                |
| <i>CONF_NUM</i>   | Numeric | 32       |         | Identifier for message to delete  |
| <i>VERSION</i>    | Numeric | 1        | 1       | 1= compatible with IP Gateway Version 3<br>2=returns additional information |

**Table 7-1 -- Required Parameters**

| Name        | Type    | Max Size | Description                             |
|-------------|---------|----------|---|
| RESULT      | Numeric | 2        | Contains error code if operation failed |
| EXTEND_DESC | String  | 1024     | Extended result description             |

**Table 7-2 -- Return Values**

| RESULT | Description   |
|--------|---|
| -1     | Invalid Session ID                                    |
| 1      | Success   |
| 10     | PARAM_FORMAT_ERROR                                    |
| 20     | FORM_VARIABLE_MISSING                                 |
| 30     | DB_ERROR  |
| 40     | DB_LOOKUP_FAILED                                      |
| 50     | INTERNAL_SERVLET_ERROR                                |
| 60     | OMS_COMM_ERROR<br>( <i>VERSION</i> =2 only)           |
| 70     | DELETE_FAILED_ACTIVE_MSG<br>( <i>VERSION</i> =2 only) |

**Table 7-3 -- Result Codes**

| EXTEND_DESC  |
|--|
| Success  |
| Session Invalid  |
| The required parameter "SESSION_ID" was not present in the form.   |
| Couldn't retrieve status for message with confNum: <i>n</i>  |
| No Rows affected by update   |
| Unable to communicate with OMS on GCC( <i>n</i> )<br>( <i>VERSION=2</i> only; <i>n</i> indicates GCC ID) |
| Record not found in OMS DB at GCC( <i>n</i> )<br>( <i>VERSION=2</i> only)                                |
| Unable to delete message due to active session at OMS on GCC( <i>n</i> )<br>( <i>VERSION=2</i> only)     |

**Table 7-4 -- Extended Result Codes**

## Chapter 8 QueryDeviceStatus

This function is used to query status of a specific SC provisioned on the OMS. The command returns the last time an ORBCOMM mobile originated message was received by the gateway from the SC in addition to other parameters described below.

The AUTHENTICATE command must be called before the QUERYDEVICESTATUS to obtain the required *SESSION\_ID* value. If the response to QUERYDEVICESTATUS is “-1, Session Invalid,” then the application must re-authenticate as the *SESSION\_ID* has expired or has been invalidated.

### Example 1 (without *VERSION* parameter):

```
http://host/QueryDeviceStatus?SESSION_ID=2231112123701635&DEVICE_ID=sc1
```

XML results:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <QUERYDEVICESTATUS>
  <RESULT>1</RESULT>
  <EXTEND_DESC>Success</EXTEND_DESC>
  <LAST_MSG_TIME>2005-01-13 20:46:56</LAST_MSG_TIME>
  <LAST_GGRAM_TIME>2005-01-13 20:46:56</LAST_GGRAM_TIME>
  <LAST_REPORT_TIME>2005-01-13 20:46:56</LAST_REPORT_TIME>
  <LAST_UPDATE_TIME>2005-03-29 12:00:00</LAST_UPDATE_TIME>
</QUERYDEVICESTATUS>
```

The QUERYDEVICESTATUS function has been enhanced in IP Gateway Version 4.0 to include the *VERSION* parameter. When *VERSION*=2 is specified in the request, additional information is returned in the response. When *VERSION*=1 or none is specified in the request, the response is compatible to IP Gateway Version 3.

### Example 2 (with *VERSION* parameter specified):

```
http://host/QueryDeviceStatus?SESSION_ID=2231112123701635&DEVICE_ID=sc1&VERSION=2
```

XML results:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <QUERYDEVICESTATUS>
  <RESULT>1</RESULT>
  <EXTEND_DESC>Success</EXTEND_DESC>
  <DEVICE_ID>sc1</DEVICE_ID>
  <GCC_ID>1</GCC_ID>
  <LAST_MSG_TIME>2005/01/13 20:46:56</LAST_MSG_TIME>
  <LAST_GGRAM_TIME>2005/01/13 20:46:56</LAST_GGRAM_TIME>
  <LAST_REPORT_TIME>2005/01/13 20:46:56</LAST_REPORT_TIME>
  <LAST_UPDATE_TIME>2005/03/29 12:00:00</LAST_UPDATE_TIME>
</QUERYDEVICESTATUS>
```

| Name              | Type    | Max Size | Default Values | Description  |
|-------------------|---------|----------|----------------|--|
| <i>SESSION_ID</i> | Numeric | 32       |                | Session identifier obtained via AUTHENTICATE                                     |
| <i>DEVICE_ID</i>  | String  | 128      |                | SC alias provisioned at the IP Gateway   |
| <i>VERSION</i>    | Numeric | 1        | 1              | 1=backward compatible to IP Gateway Version 3<br>2=return additional information |

**Table 8-1 -- Request Parameters**

| Name             | Type    | Max Size | Description   |
|------------------|---------|----------|---|
| RESULT           | Numeric | 2        | Contains error code if operation failed   |
| EXTEND_DESC      | String  | 1024     | Extended result description   |
| DEVICE_ID        | String  | 128      | If <i>VERSION</i> =2, the SC alias of the device provisioned at the IP Gateway is returned  |
| GCC_ID           | Numeric | 1        | If <i>VERSION</i> =2, GCC_ID is returned with one of the following values:<br>1 = SC provisioned on US GCC<br>2 = SC provisioned on South America GCC<br>3 = SC provisioned on Europe GCC<br>4 = SC provisioned on Japan GCC<br>5 = SC provisioned on Korea GCC<br>6 = SC provisioned on Malaysia GCC |
| LAST_MSG_TIME    | String  | 19       | Last time a message was received from device<br>If <i>VERSION</i> =1 (and by default), the format is YYYY-MM-DD hh:mm:ss (GMT)<br>If <i>VERSION</i> =2, the format is YYYY/MM/DD hh:mm:ss (GMT)   |
| LAST_REPORT_TIME | String  | 19       | Last time a report was received from device<br>If <i>VERSION</i> =1 (and by default), the format is YYYY-MM-DD hh:mm:ss (GMT)<br>If <i>VERSION</i> =2, the format is YYYY/MM/DD hh:mm:ss (GMT).   |
| LAST_UPDATE_TIME | String  | 19       | Last time the OMS record for this SC changed<br>If <i>VERSION</i> =1 (and by default), the format is YYYY-MM-DD hh:mm:ss (GMT)<br>If <i>VERSION</i> =2, the format is YYYY/MM/DD hh:mm:ss (GMT)   |

**Table 8-2 -- QUERYDEVICESTATUS Return Values**

| RESULT | Description                              |
|--------|--|
| -1     | Invalid Session ID                       |
| 1      | Success                                  |
| 10     | PARAM_FORMAT_ERROR                       |
| 20     | FORM_VARIABLE_MISSING                    |
| 30     | DB_ERROR                                 |
| 40     | DB_LOOKUP_FAILED                         |
| 50     | INTERNAL_SERVLET_ERROR                   |
| 60     | OMS_DB_ERROR<br>(VERSION=2 only)         |
| 70     | OMS_DB_LOOKUP_FAILED<br>(VERSION=2 only) |

**Table 8-3 -- QUERYDEVICESTATUS Result Codes**

| EXTEND_DESC  |
|--|
| Session Invalid  |
| Invalid DEVICE_ID  |
| IMD Query failed<br>(VERSION=1 only)   |
| The required parameter "SESSION_ID" was not present in the form                        |
| The required parameter "DEVICE_ID" was not present in the form                         |
| OMS Query has failed at GCC( <i>n</i> )<br>(VERSION=2 only; <i>n</i> indicates GCC ID) |
| Device not provisioned in OMS DB at GCC ( <i>n</i> )<br>(VERSION=2 only)               |

**Table 8-4 -- QUERYDEVICESTATUS Extended Result Codes**

## Chapter 9 RETRIEVEMESSAGES

This function is used to retrieve SC-originated and SC-terminated messages, based on the selection criteria. All messages that are associated with a given user profile, can be retrieved with a RETRIEVEMESSAGES request. Messages can also be retrieved based on various criteria, as detailed below. Additionally, the MESSAGE\_FLAG parameter may be set for multiple messages using RETRIEVEMESSAGES.

AUTHENTICATE must be called before RETRIEVEMESSAGES to obtain the required SESSION\_ID value. If the response to RETRIEVEMESSAGES is “-1, Session Invalid,” then the application must re-authenticate as the SESSION\_ID has expired or has been invalidated.

### Example 1 (without VERSION parameter):

```
http://host/RetrieveMessages?SESSION_ID=2231112123701635&NETWORK_ID=0&MSG_FLAG=0&SET_FLAG=0&MSG_STATUS=0&MESSAGE_ID=0&MESSAGE=0&MTAG=0
```

### XML result:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <RETRIEVEMESSAGES>
  <CONF_NUM>200000014</CONF_NUM>
  <MESSAGE_ID>14</MESSAGE_ID>
  <MESSAGE_TIME>2005/03/22 19:06:39</MESSAGE_TIME>
  <MESSAGE_FROM>test1</MESSAGE_FROM>
  <MESSAGE_TO>sc1</MESSAGE_TO>
  <MESSAGE_SUBJECT>test to sc1</MESSAGE_SUBJECT>
  <MESSAGE_ENCODING>ASCII</MESSAGE_ENCODING>
  <MESSAGE_FLAG>READ</MESSAGE_FLAG>
  <MESSAGE_STATUS>Delivered</MESSAGE_STATUS>
  <NETWORK_ID>3</NETWORK_ID>
  <MESSAGE_DIRECTION>Mobile Terminated</MESSAGE_DIRECTION>
  <CONF_NUM>200000026</CONF_NUM>
  <MESSAGE_ID>26</MESSAGE_ID>
  <MESSAGE_TIME>2005/03/23 18:26:21</MESSAGE_TIME>
  <MESSAGE_FROM>test1</MESSAGE_FROM>
  <MESSAGE_TO>sc2</MESSAGE_TO>
  <MESSAGE_SUBJECT>test to sc2</MESSAGE_SUBJECT>
  <MESSAGE_ENCODING>ASCII</MESSAGE_ENCODING>
  <MESSAGE_FLAG>READ</MESSAGE_FLAG>
  <MESSAGE_STATUS>Delivered</MESSAGE_STATUS>
  <NETWORK_ID>3</NETWORK_ID>
  <MESSAGE_DIRECTION>Mobile Terminated</MESSAGE_DIRECTION>
  <CONF_NUM>NO_CONF_NUM</CONF_NUM>
  <MESSAGE_ID>28</MESSAGE_ID>
  <MESSAGE_TIME>2005/03/23 20:26:40</MESSAGE_TIME>
  <MESSAGE_FROM>sc1</MESSAGE_FROM>
  <MESSAGE_TO>test1</MESSAGE_TO>
  <MESSAGE_SUBJECT>Testing 2 to new ipgwy</MESSAGE_SUBJECT>
  <MESSAGE_ENCODING>ASCII</MESSAGE_ENCODING>
  <MESSAGE_FLAG>READ</MESSAGE_FLAG>
```

```
<MESSAGE_STATUS>SC Originated</MESSAGE_STATUS>
<NETWORK_ID>3</NETWORK_ID>
<MESSAGE_DIRECTION>Mobile Originated</MESSAGE_DIRECTION>
  <CONF_NUM>100000029</CONF_NUM>
<MESSAGE_ID>29</MESSAGE_ID>
<MESSAGE_TIME>2005/03/23 20:58:45</MESSAGE_TIME>
<MESSAGE_FROM>test1</MESSAGE_FROM>
<MESSAGE_TO>scl</MESSAGE_TO>
<MESSAGE_SUBJECT>test to scl msg2</MESSAGE_SUBJECT>
<MESSAGE_ENCODING>ASCII</MESSAGE_ENCODING>
<MESSAGE_FLAG>READ</MESSAGE_FLAG>
<MESSAGE_STATUS>Delivered</MESSAGE_STATUS>
<NETWORK_ID>3</NETWORK_ID>
<MESSAGE_DIRECTION>Mobile Terminated</MESSAGE_DIRECTION>
<RESULT>1</RESULT>
<EXTEND_DESC>Success</EXTEND_DESC>
</RETRIEVEMESSAGES>
```

The RETRIEVEMESSAGES function has been enhanced in IP Gateway Version 4.0 to include the VERSION parameter. When VERSION=2 is specified in the request, up to 500 messages are returned in the response in descending order from newest to oldest. Additional information is also returned when VERSION is set to 2. When VERSION=1 or none is specified in the request, the response is compatible to IP Gateway Version 3.x. When VERSION is set to 1, up to 200 messages are returned in ascending order from oldest to newest.

Note that when MESSAGE\_ID is specified, the results will include the set of messages with the lowest MESSAGE\_TIME values meeting the other criteria and having a MESSAGE\_TIME greater than or equal to the message with the specified ID. For example, under version 2 if MESSAGE\_ID of 500 is specified, the results will include the next 500 messages for the account with message times greater than that of the message with ID 500, including that message if it meets the other criteria, in order from newest to oldest. These will not necessarily include the newest messages: if there are more than 500 messages which otherwise meet the criteria, those with the lowest message times will be returned. To retrieve the next set of messages, the RetrieveMessages command would be re-issued using the value of the MESSAGE\_ID of the latest message retrieved from the previous result set as the MESSAGE\_ID value in the new command.

Table 9-1 summarizes the results that can be expected for various values of MESSAGE\_ID for each API version.



| API Version | MESSAGE_ID value | Result   |
|-------------|------------------|--|
| 1           | -1               | Up to 200 oldest messages meeting other criteria   |
|             | 0                | Same as -1   |
|             | <i>n</i>         | Up to 200 oldest messages with MESSAGE_TIME later than or equal to the message with MESSAGE_ID <i>n</i> meeting other criteria |
| 2           | -1               | Up to 500 newest messages meeting other criteria   |
|             | 0                | Up to 500 oldest messages meeting other criteria   |
|             | <i>n</i>         | Up to 500 oldest messages with MESSAGE_TIME later than or equal to the message with MESSAGE_ID <i>n</i> meeting other criteria |

Table 9-1 -- Effect of MESSAGE\_ID on RETRIEVEMESSAGES results

**Example 2 (with VERSION parameter specified):**

```
http://host/RetrieveMessages?SESSION_ID=2231112123701635&NETWORK_ID=0&MSG_FLAG=0&SET_FLAG=0&MSG_STATUS=0&MESSAGE_ID=0&MESSAGE=0&MTAG=0&VERSION=2
```

**XML result:**

```
<?xml version="1.0" encoding="UTF-8" ?>
- <RETRIEVEMESSAGES>
  <CONF_NUM>100000029</CONF_NUM>
  <MESSAGE_ID>29</MESSAGE_ID>
  <MESSAGE_TIME>2005/03/23 20:58:45</MESSAGE_TIME>
  <MESSAGE_FROM>test1</MESSAGE_FROM>
  <MESSAGE_TO>sc1</MESSAGE_TO>
  <MESSAGE_SUBJECT>test to sc1 msg2</MESSAGE_SUBJECT>
  <MESSAGE_ENCODING>ASCII</MESSAGE_ENCODING>
  <MESSAGE_FLAG>READ</MESSAGE_FLAG>
  <MESSAGE_STATUS>Delivered</MESSAGE_STATUS>
  <NETWORK_ID>3</NETWORK_ID>
  <MESSAGE_DIRECTION>Mobile Terminated</MESSAGE_DIRECTION>
  <CONF_NUM>NO_CONF_NUM</CONF_NUM>
  <MESSAGE_ID>28</MESSAGE_ID>
  <MESSAGE_TIME>2005/03/23 20:26:40</MESSAGE_TIME>
  <MESSAGE_FROM>sc1</MESSAGE_FROM>
  <MESSAGE_TO>test1</MESSAGE_TO>
  <MESSAGE_SUBJECT>Testing 2 to new ipgwy</MESSAGE_SUBJECT>
  <MESSAGE_ENCODING>ASCII</MESSAGE_ENCODING>
  <MESSAGE_FLAG>READ</MESSAGE_FLAG>
  <MESSAGE_STATUS>SC Originated</MESSAGE_STATUS>
  <NETWORK_ID>3</NETWORK_ID>
  <MESSAGE_DIRECTION>Mobile Originated</MESSAGE_DIRECTION>
  <CONF_NUM>200000026</CONF_NUM>
  <MESSAGE_ID>26</MESSAGE_ID>
  <MESSAGE_TIME>2005/03/23 18:26:21</MESSAGE_TIME>
  <MESSAGE_FROM>test1</MESSAGE_FROM>
  <MESSAGE_TO>sc2</MESSAGE_TO>
  <MESSAGE_SUBJECT>test to sc2</MESSAGE_SUBJECT>
```

```
<MESSAGE_ENCODING>ASCII</MESSAGE_ENCODING>
<MESSAGE_FLAG>READ</MESSAGE_FLAG>
<MESSAGE_STATUS>Delivered</MESSAGE_STATUS>
<NETWORK_ID>3</NETWORK_ID>
<MESSAGE_DIRECTION>Mobile Terminated</MESSAGE_DIRECTION>
<CONF_NUM>200000014</CONF_NUM>
<MESSAGE_ID>14</MESSAGE_ID>
<MESSAGE_TIME>2005/03/22 19:06:39</MESSAGE_TIME>
<MESSAGE_FROM>test1</MESSAGE_FROM>
<MESSAGE_TO>sc1</MESSAGE_TO>
<MESSAGE_SUBJECT>test to sc1</MESSAGE_SUBJECT>
<MESSAGE_ENCODING>ASCII</MESSAGE_ENCODING>
<MESSAGE_FLAG>READ</MESSAGE_FLAG>
<MESSAGE_STATUS>Delivered</MESSAGE_STATUS>
<NETWORK_ID>3</NETWORK_ID>
<MESSAGE_DIRECTION>Mobile Terminated</MESSAGE_DIRECTION>
<RESULT>1</RESULT>
<EXTEND_DESC>Success</EXTEND_DESC>
</RETRIEVEMESSAGES>
```

| Name   | Type    | Max Size | Default Values | Description   |
|--|---------|----------|----------------|---|
| SESSION_ID   | Numeric | 32       |                | Session Identifier obtained via Authenticate  |
| NETWORK_ID   | Numeric | 2        | 3              | 3 = ORBCOMM   |
| MSG_FLAG<br>(Used to filter results by current flag setting) | Numeric | 1        | 0              | 0 = ALL (READ, UNREAD, DELETED)<br>1 = READ<br>2 = UNREAD<br>3 = DELETED  |
| SET_FLAG<br>(Used to change the flag setting)                | Numeric | 1        | 0              | 0 = No Action (MSG_FLAG does not change)<br>1 = READ<br>2 = UNREAD<br>3 = DELETED   |
| MSG_STATUS<br>(Used to filter results)                       | Numeric | 2        | 0              | 0 = ALL<br>1 = PENDING<br>2 = RECEIVED_OMS (VERSION=2 only)<br>3 = DELIVERED<br>4 = IN TRANSIT<br>5 = SCHEDULED<br>6 = MOBILE ORIGINATED<br>7 = EXHAUSTED (VERSION=2 only)<br>9 = UNDELIVERABLE |
| MESSAGE_ID   | Numeric | 1        | 0              | Valid Message ID or -1 to disable indexing  |
| MESSAGE  | Numeric | 1        | 0              | 0 = Do not return Message Body<br>1 = Return Message Body   |
| MTAG   | Numeric | 1        | 0              | 0 = RetrieveMessages is XML parent<br>1 = Message is XML parent (each message is wrapped in MESSAGE tag for XML parsing of multiple messages)   |
| VERSION  | Numeric | 1        | 1              | 1=backward compatible to IP Gateway Version 3<br>2=return additional information  |

Table 9-2 -- RETRIEVEMESSAGES Parameters

| Name                  | Type    | Max Size | Description   |
|-----------------------|---------|----------|---|
| CONF_NUM              | Numeric | 32       | Confirmation number of mobile terminated message  |
| MESSAGE_ID            | Numeric | 32       | Message ID of message   |
| SMTP_MSG_ID           | String  | 128      | If <i>VERSION=2</i> , this is the unique identifier for a message. If <i>MESSAGE_STATUS</i> indicates "Pending" or "Delayed", this field is null.   |
| GCC_ID                | Numeric | 1        | If <i>VERSION=2</i> , GCC_ID is returned with one of the following values:<br>1 = SC provisioned on US GCC<br>2 = SC provisioned on South America GCC<br>3 = SC provisioned on Europe GCC<br>4 = SC provisioned on Japan GCC<br>5 = SC provisioned on Korea GCC<br>6 = SC provisioned on Malaysia GCC |
| MESSAGE_PRIORITY      | String  | String   | If <i>VERSION=2</i> , priority of message is returned with one of the following: non-urgent; normal; urgent   |
| MESSAGE_TIME          | String  | 19       | SCT-Time message was created at the IP Gateway<br>SCO-Time message was received at the OMS  |
| MESSAGE_FROM          | String  | 128      | Originator address  |
| MESSAGE_TO            | String  | 128      | Destination address   |
| MESSAGE_SUBJECT       | String  | 128      | Subject of the message  |
| MESSAGE_ENCODING      | String  | 6        | Body encoding type (ASCII/Binary)   |
| MESSAGE_BODY          | String  | 2000     | Body of the message   |
| MESSAGE_FLAG          | String  | 16       | Text describing flagged state of message  |
| MESSAGE_STATUS        | String  | 16       | Text describing status of message   |
| DELIVERED_FAILED_TIME | String  | 19       | <i>VERSION=2</i> only.<br>SCT-Time when message was successfully delivered to SC or failed. If message is not yet completed, this value indicates when the message was stored.<br>SCO-Time message was received at the IP Gateway   |
| MESSAGE_DIRECTION     | String  | 18       | "Mobile Originated" or "Mobile Terminated"  |
| NETWORK_ID            | Numeric | 2        | 3 = ORBCOMM   |
| RESULT                | Numeric | 2        | Contains error code if operation failed   |

Table 9-3 -- RETRIEVE\_MESSAGES Return Values

| MESSAGE_STATUS | Description   |
|----------------|---|
| Pending        | The message has been accepted by the IP Gateway for delivery  |
| InTransit      | The message has been accepted by the network for delivery   |
| Scheduled      | The message is being held by the IP Gateway for delivery until the specified time that was requested in the delivery send_time option |
| ReceivedOMS    | The message is received by the OMS waiting to be delivered to the SC (VERSION=2 only)   |
| Exhausted      | The OMS has exhausted all attempts to deliver the message to the SC (VERSION=2 only)  |
| Delivered      | The message was delivered   |
| Undeliverable  | The message has been rejected or cannot be delivered  |

**Table 9-4 -- RETRIEVE\_MESSAGES Message Status codes**

| RESULT | Description            |
|--------|------------------------|
| -1     | Invalid Session ID     |
| 1      | Success                |
| 10     | PARAM_FORMAT_ERROR     |
| 20     | FORM_VARIABLE_MISSING  |
| 30     | DB_ERROR               |
| 40     | DB_LOOKUP_FAILED       |
| 50     | INTERNAL_SERVLET_ERROR |

**Table 9-5 -- RETRIEVE\_MESSAGES Result Codes**

| EXTEND_DESC  |
|--|
| Success  |
| Session Invalid  |
| The required parameter "SESSION_ID" was not present in the form. |
| Illegal value (n) for NETWORK_ID                                 |
| Illegal value (n) for MSG_FLAG                                   |
| Illegal value (n) for MARK_FLAG                                  |
| Illegal value (n) for MSG_STATUS                                 |
| Illegal value (n) for MESSAGE                                    |
| No Messages Meet the Selection Criteria*                         |

**Table 9-6 -- RETRIEVE\_MESSAGES Extended Result Codes**

\* If no messages meet the selection criteria the EXTEND\_DESC response will contain the words "No Messages Meet the Selection Criteria" with a RESULT of 1 (Success).

## Chapter 10 SETMESSAGEFLAG

This command is used to specify the message flag for a particular message or set of messages. Message retrieval can be performed based on message flags, allowing results to be filtered (see Chapter 9). The **AUTHENTICATE** command must be called before the **SETMESSAGEFLAG** command to obtain the required *SESSION\_ID* value. If the response to **SETMESSAGEFLAG** is -1, Session Invalid, then the application must re-authenticate as the *SESSION\_ID* has expired or has been invalidated.

### Example:

```
http://host/SetMessageFlag?SESSION_ID=2351112208299111&SELECT=1&CRITERIA=200003311000&FLAG=1
```

### XML results

```
<?xml version="1.0" encoding="UTF-8" ?>
<SETMESSAGEFLAG>
  <MESSAGES>1288</MESSAGES>
  <RESULT>1</RESULT>
  <EXTEND_DESC>Success</EXTEND_DESC>
</SETMESSAGEFLAG>
```

| Name              | Type    | Max Size | Description  |
|-------------------|---------|----------|--|
| <i>SESSION_ID</i> | Numeric | 32       | Session identifier obtained via <b>AUTHENTICATE</b>  |
| <i>SELECT</i>     | Numeric | 1        | 0 = ALL<br>1 = BY DATE<br>2 = BY CONF_NUM<br>3 = BY MESSAGE_ID   |
| <i>CRITERIA</i>   | String  | 32       | Criteria contains data based on <b>SELECT</b> option:<br>0: No data required<br>1: Date (YYYYMMDDHHMM)<br>2: CONF_NUM<br>3: MESSAGE_ID |
| <i>FLAG</i>       | Numeric | 1        | 1 = READ<br>2 = UNREAD<br>3 = DELETED  |

Table 10-1 -- SETMESSAGEFLAG Required Parameters

| Name        | Type    | Max Size | Description                             |
|-------------|---------|----------|---|
| MESSAGES    | Numeric | 4        | Count of messages updated               |
| RESULT      | Numeric | 2        | Contains error code if operation failed |
| EXTEND_DESC | String  | 1024     | Extended result description             |

Table 10-2 -- SETMESSAGEFLAG Return Values

| RESULT | Description            |
|--------|------------------------|
| -1     | Invalid Session ID     |
| 1      | Success                |
| 10     | PARAM_FORMAT_ERROR     |
| 20     | FORM_VARIABLE_MISSING  |
| 30     | DB_ERROR               |
| 40     | DB_LOOKUP_FAILED       |
| 50     | INTERNAL_SERVLET_ERROR |

**Table 10-3 -- SETMESSAGEFLAG Result Codes**

| EXTEND_DESC  |
|--|
| Success  |
| Session Invalid  |
| Illegal value (n) for SELECT                                     |
| Illegal value (n) for CRITERIA                                   |
| The required parameter "SESSION_ID" was not present in the form. |
| No Rows affected by update                                       |

**Table 10-4 -- SETMESSAGEFLAG Extended Result Codes**

## Chapter 11 HTTP Notification

The HTTP notification service allows customers to receive notifications, in HTTP format, that messages have arrived in the IP Gateway. Notifications are only possible for SC-originated messages. The notification utilizes an HTTP POST command to the URL specified by the application developer during the authentication process.

The output of each notification will be an HTTP POST that has a single parameter-value pair. The parameter is "MSG" with a value representing the number of messages that have arrived at the IP Gateway for the account.

This notification service makes the application more efficient by reducing and or eliminating the need to issue repeated RETRIEVEMESSAGES commands to the IP Gateway. Such commands need only be issued to check for SC-originated messages after receipt of a notification.

The AUTHENTICATE command has an additional parameter that is required to activate the notification function. This parameter is in the form of a URL. Specifying a URL will register that session to receive notifications of SC-originated messages. For accounts simultaneously logged in to multiple sessions, only the URL from the last AUTHENTICATE command will receive notifications.

### Example:

```
http://host/Authenticate?LOGIN=xxxx&PSSWD=yyyy&URL=http://11.12.13.14/
Notify
```

### XML result:

```
<?xml version="1.0" encoding="UTF-8" ?>
<AUTHENTICATE>
  <SESSION_ID>2351112208299111</SESSION_ID>
  <RESULT>1</RESULT>
  <EXTEND_DESC>Success</EXTEND_DESC>
</AUTHENTICATE>
```

The following are the results of an example and the expected results that appear in DOS window indication a notification that a message has arrived to the IP Gateway.

```
ServerSocket[addr=0.0.0.0/0.0.0.0,port=0,localport=80]
Waiting for connection
Got connection
from:Socket[addr=11.12.13.14/11.12.13.14,port=62527,localport=80
]
POST /notify HTTP/1.0
Content-Type: application/x-www-form-urlencoded
User-Agent: Java1.1.6
Host: 11.12.13.14
Accept: text/html, image/gif, image/jpeg, */*; q=.2, */*; q=.2
Content-length: 5

MSG=1
```

## Chapter 12 REFRESH

The REFRESH command is new to IP Gateway Version 4.0. It is used to refresh the current session. This is a simple method to keep a session alive when there is no activity. All other API transactions serve to refresh the session as well, and no explicit use of the REFRESH command if commands are being regularly sent in the course of business.

### Example:

`http://host/Refresh?SESSION_ID=2351112208299111`

### XML results:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <REFRESH>
  <RESULT>1</RESULT>
  <EXTEND_DESC>Success</EXTEND_DESC>
  <SESS_EXPIRE_TIME>2005/03/30 19:16:29</SESS_EXPIRE_TIME>
  <TOTAL_ACTIVE_SESS>1</TOTAL_ACTIVE_SESS>
</REFRESH>
```

| Name       | Type    | Max Size | Description                                  |
|------------|---------|----------|--|
| SESSION_ID | Numeric | 32       | Session Identifier obtained via AUTHENTICATE |

Table 12-1 -- REFRESH Parameter

| Name              | Type    | Max Size | Description   |
|-------------------|---------|----------|---|
| RESULT            | String  | 2        | Contains error code if operation failed               |
| EXTEND_DESC       | String  | 1024     | Verbose error description                             |
| SESS_EXPIRE_TIME  | String  | 19       | Time the session will expire                          |
| TOTAL_ACTIVE_SESS | Numeric | 3        | Total number of active sessions for the login account |

Table 12-2 -- REFRESH Return Values



| RESULT | Description            |
|--------|------------------------|
| -1     | Invalid Session ID     |
| 1      | Success                |
| 10     | PARAM_FORMAT_ERROR     |
| 20     | FORM_VARIABLE_MISSING  |
| 30     | DB_ERROR               |
| 50     | INTERNAL_SERVLET_ERROR |

**Table 12-3 -- REFRESH Result Codes**

| EXTEND_DESC  |
|--|
| Success  |
| Session Invalid  |
| The required parameter "SESSION_ID" was not present in the form. |
| Database error   |
| Internal software error  |

**Table 12-4 -- REFRESH Extended Result Codes**

## Chapter 13 LOGOUT

The LOGOUT command is used to logout of the current session.

**Example:**

```
http://host/Logout?SESSION_ID=2351112208299111
```

**XML results:**

```
<?xml version="1.0" encoding="UTF-8" ?>
- <LOGOUT>
  <RESULT>1</RESULT>
</LOGOUT>
```

| Name       | Type    | Max Size | Description                                  |
|------------|---------|----------|--|
| SESSION_ID | Numeric | 32       | Session Identifier obtained via Authenticate |

**Table 13-1 -- LOGOUT Parameter**

| Name   | Type    | Max Size | Description                      |
|--------|---------|----------|----------------------------------|
| RESULT | Numeric | 2        | Contains error code of operation |

**Table 13-2 -- LOGOUT Return Value**

| RESULT | Description           |
|--------|-----------------------|
| 1      | Success               |
| 20     | FORM_VARIABLE_MISSING |

**Table 13-3 -- LOGOUT Result Codes**

## Appendix A– HTTP URL Encoding Scheme

The HTTP URL scheme is used to designate Internet resources accessible using HTTP (HyperText Transfer Protocol) and is used as the interface to the IP Gateway relay. This only describes the syntax of URLs to be used to access the developer relay. An HTTP URL takes the form:

<http://<host>:<port>/<path>?<searchpart>>

Where <host> and <port> are as described above. If :<port> is omitted, the port defaults to 80. <path> is an HTTP selector, and <searchpart> is a query string. The <path> is optional, as is the <searchpart> and its preceding "?". If neither <path> nor <searchpart> is present, the "/" may also be omitted. Within the <path> and <searchpart> components, "/", ";", "?" are reserved. The "/" character may be used within HTTP to designate a hierarchical structure.

URLs are sequences of characters, i.e., letters, digits, and special characters. A URL may be represented as a sequence of octets in a coded character set. The interpretation of a URL depends only on the identity of the characters used. In most URL schemes, the sequences of characters in different parts of a URL are used to represent sequences of octets used in Internet protocols. Octets may be encoded by a character triplet consisting of the character "%" followed by the two hexadecimal digits (from "0123456789ABCDEF") which forming the hexadecimal value of the octet. (The characters "abcdef" may also be used in hexadecimal encodings.) Octets must be encoded if they have no corresponding graphic character within the US-ASCII coded character set, if the use of the corresponding character is unsafe, or if the corresponding character is reserved for some other interpretation within the particular URL scheme.

URLs are written only with the graphic printable characters of the US-ASCII coded character set. The octets 80-FF hexadecimal are not used in US-ASCII, and the octets 00-1F and 7F hexadecimal represent control characters; these must be encoded. Unsafe Characters can be unsafe for a number of reasons. The space character is unsafe because significant spaces may disappear and insignificant spaces may be introduced when URLs are transcribed or typeset or subjected to the treatment of word-processing programs. The characters "<" and ">" are unsafe because they are used as the delimiters around URLs in free text; the quote mark ("") is used to delimit URLs in some systems. The character "#" is unsafe and should always be encoded because it is used in World Wide Web and in other systems to delimit a URL from a fragment/anchor identifier that might follow it. The character "%" is unsafe because it is used for encoding of other characters. Other characters are unsafe because gateways and other transport agents are known to sometimes modify such characters. These characters are "{", "}", "|", "\", "^", "~", "[", "]", and "'". All unsafe characters must always be encoded within a URL. For example, the character "#" must be encoded within URLs even in systems that do not normally deal with fragment or anchor identifiers, so that if the URL is copied into another system that does use them, it will not be necessary to change the URL. Many URL schemes reserve certain characters for a special meaning: their

appearance in the scheme-specific part of the URL has a designated semantics. If the character corresponding to an octet is reserved in a scheme, the octet must be encoded. The characters ";", "/", "?", ":", "@", "=", and "&" are the characters which may be reserved for special meaning within a scheme.

## Appendix B- XML Response Format

Extensible Markup Language (XML) describes a class of data objects called XML documents. XML documents are made up of storage units called entities, which contain either parsed or unparsed data. Parsed data is made up of characters, some of which form character data, and some of which form markup. Markup encodes a description of the document's storage layout and logical structure.

A software module called an XML processor is used to read XML documents and provide access to their content and structure. Responses from developer relay functions will be in XML format allowing the client applications to use an XML processor. This frees the client application from the requirement of parsing the responses and also allows data flow into applications such as a web browser which will allow formatting and or further processing with simple scripting languages.

The XML specification is available at <http://www.w3.org>.

## Appendix C– Frequently Asked Questions

**1. How many messages per customer can be stored on the server?**

No limit is defined in software per customer. Messages are deleted after a maximum of 20 days, with shorter periods available via provisioning.

**2. When a SC-terminated message is sent from the host site with a delivery time, what is the accuracy of the message being delivered to the subscriber unit at the designated time? Within so many minutes/seconds?**

The gateway delivers the message to ORBCOMM Message Switch (OMS) at the specified delivery time within a few seconds. The rest is up to the satellite network.

**3. When are messages time stamped?**

The MESSAGE\_TIME parameter is set for SC-originated messages when they are received over the satellites network at the OMS. For an SC-terminated message, MESSAGE\_TIME is set when the message is received by the IP Gateway from the host application.

API version 2 (IP Gateway version 4) also associates a parameter called DELIVERED\_FAILED\_TIME with each message. For SC-originated messages, this is set to the time the message arrives at the IP Gateway. For SC-terminated messages, it is the time the message was delivered or marked undeliverable. If the SC-terminated message is neither yet completed nor failed, the time is the same as MESSAGE\_TIME.

**4. What kind of ability does ORBCOMM have to track mobile-to-mobile communication when the IP Gateway services the customer?**

You can't get status of mobile-to-mobile messages because they never go through the IP Gateway. The OMS captures this data and generates billing records as normal, but no record is passed to the IP Gateway.

**5. We get "No Confirmation number" from the messages that the ORBCOMM radio has sent. Why is that?**

The confirmation number is typically used by a host application to request message status. Since the host application would not know the confirmation number of an SC-originated message until the message is received by the IP Gateway (and thus available for retrieval from a RETRIEVEMESSAGES command), we do not see the value in a confirmation number for SC-originated messages.

6. **According to the spec, we are not supposed to access the server more than once a minute. This means that if we have 10 messages to send we can send only one every minute and that we then cannot check messages during this time, or we would slow down the sending even more. Should it be like this?**

No. Please send messages frequently. The one-minute guideline is for RETRIEVEMESSAGES.

Note also that if a RETRIEVEMESSAGES command results in the maximum number of messages being returned, we expect that a subsequent RETRIEVEMESSAGES command will be issued immediately to ensure that all messages available are retrieved, rather than waiting for a minute. This presumes, naturally, that the query being executed is limited to retrieving newly arrived messages, rather than retrieving every message sent and received in the last 20 days every 60 seconds. The MSG\_FLAG parameter can be used to make limit those queries.

7. **How does the user profile work?**

At the time of provisioning each customer will select a username/password and have subscriber communicators associated with that username. When a host application goes through the AUTHENTICATE process for a particular username and then issues a RETRIEVEMESSAGE command, the IP Gateway will respond with the messages from all the radios registered to that username. A username will not receive messages for other accounts, nor be able to send messages to another account's SCs.

8. **Is there a time-out for active sessions?**

If there is no activity the session will terminate after 30 minutes. If commands are sent at the maximum allowable rate of one per minute, the session should not expire. The REFRESH command is also available in API version 2 to maintain a session. Of course, having a mechanism to automatically restart the host-initiated connection if the session becomes invalid is highly recommended, as sessions may occasionally be interrupted.

9. **Will the "retrieve message" mechanism replace or simple augment SMTP-based delivery?**

The IP Gateway is not meant to replace the ORBCOMM network access option of SMTP. It is anticipated that personal messaging applications will continue to use SMTP email, but business applications that require additional security; reliability and features will prefer the HTTP/XML interface offered by our new product. If a SC-originated message is sent to ipqwy@ipgwy.orbcomm.net then

it goes to the IP Gateway and can only be retrieved by a IP Gateway commands.

**10. Are there source code examples or APIs available to ease application development for interfacing via IP to ORBCOMM?**

ORBCOMM offers a Java API for developers to use to create IP Gateway applications. Contact your account representative or ORBCOMM Customer Service for more information.