inmarsat

ISATDATA PRO - MESSAGING APIS

VERSION 3.0

PUBLICATION DATE: 11-FEB-2021

Contents

1: Introduction

1.1: Purpose

This document provides an overview of the IsatData Pro Gateway Messaging API for developers.

1.2: Audience

This document is intended for developers seeking to integrate customer applications with the IsatData Pro service.

1.3: Notation

IsatData Pro Gateway Messaging API is platform independent.

A **forward**, or To-Mobile, or Mobile-Terminated message is a message sent to a terminal from the Message Gateway System (MGS), while a **return**, or From-Mobile, or Mobile-Originated message is one sent from a terminal.

1.4: Reference

It is recommended that you be familiar with the content of the following documents before using this guide. These Inmarsat documents are available on the Inmarsat Developer Portal at https://developer.inmarsat.com.

- *IsatData Pro Service Description*
- *IsatData Pro Modem Message Specification*
- *IsatData Pro Message Definition File Description*

Note: Additional information and toolkits for developing software for terminals may be available from the terminal manufacturers.

2: Solution Overview

IsatData Pro is a global L-band satellite network service optimized for Machine-to-Machine (M2M) and Internet of Things (IoT) applications. IsatData Pro is intended for event-driven data collection and remote control, but also enables applications such as forms and text messaging through Human-Machine Interfaces. Typical applications include tracking, fleet management, security, remote monitoring, telematics and SCADA.

The IsatData Pro service includes a collection of webservices that will allow its users to manage users' terminals and traffic and to exchange data messages with terminals.

This document provides a description of the IsatData Pro Messaging API Web Services for developers and DP administrators.

The main elements of Isatdata Pro (IDP) service are a Message Gateway System to manage IDP message traffic, LES equipment for the satellite communication part and Terminals in the field.

Figure 1 illustrates the various elements that are part of the IDP service and the place of the webservices.

This document focuses on the Messaging web service. See the *IDP Service Description* (see **Reference** on page 3) for a more elaborate description of the IDP service.

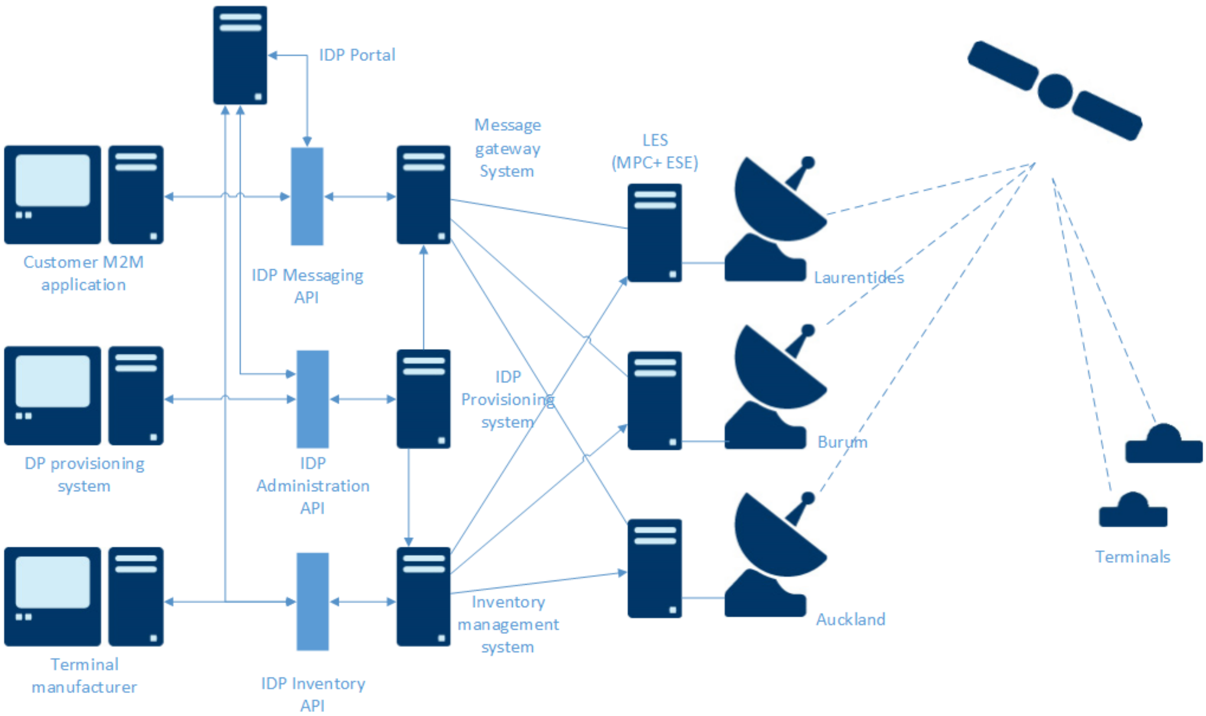

Figure 1. The IsatData Pro network overview

2.1: The IsatData Pro Messaging API

Inmarsat's IsatData Pro network provides messaging services between terminals and an Internet application. The Messaging API supports a RESTful interface. The APIs are platform- and language-independent allowing for easy integration with various IoT platforms and enterprise software.

A RESTful web service is a web service implemented using HTTP(s) and the principles of Representational State Transfer. RESTful applications use HTTP requests to post data (create and/or update), read data (e.g., make queries), and delete data. The REST API supports both JSON and XML encoding. The formal specification of the REST API is documented in a Swagger (YAML) file available from the Inmarsat Developer Portal.

2.2: Messages

All data sent via the IDP network is formed of messages of variable size. The basic message structure consists of up to 2 bytes overhead followed by a data payload.

The first byte of a message is called the Service Identification Number (SIN) which reserves a set of restricted values (0-15) as Core Modem messages for the network operator, but is otherwise usable by application developers as a means to identify message groupings for filtering and parsing. Core Modem messages are detailed in the An application developer could choose to use non-reserved SIN values as part of their message payload.

The second byte of a message is called the Message Identification Number (MIN) which is intended for use by application developers to define individual message payload structure. An application developer could choose to use MIN as part of their message payload.

Note: More detail on messaging and network operation can be found in the *IsatData Pro Service Description*.

2.3: Applications

Figure 2 shows the role of the Messaging API in the message chain.

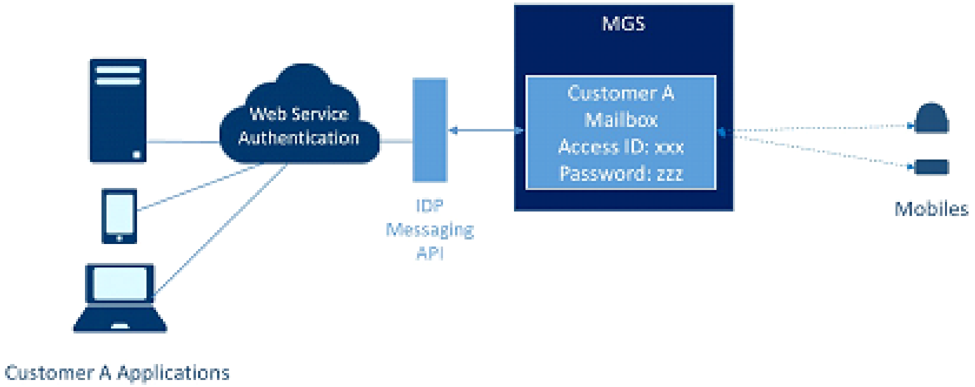

Figure 2. Messaging API in the Context of the Network

Customer Applications will typically be an IoT platform or enterprise software, but could also be a PC or mobile application talking directly to the web service. Multiple client applications can access the API/Mailbox simultaneously.

2.4: Mailbox

IsatData Pro is a store-and-forward service. To exchange messages with terminals, a Mailbox concept is used. Multiple terminals are assigned to a single mailbox, which presents a unique login credential for access to those terminals. Each Messaging API operation is authenticated against a specific mailbox's credentials. Mailboxes and credentials are created and maintained by an IsatData Pro service provider, who supplies details to an application developer. A given application might access one or more mailboxes depending on the solution architecture chosen.

Each mailbox has the following:

- Credentials in the form of login ID/password pair
- Set of mobile ID(s) associated with the Mailbox
- (optional) Set of broadcast IDs (0 or more) associated with the Mailbox
- (optional) Message definition file or codec

An authenticated Mailbox user can:

- Retrieve and submit messages from/to associated mobile IDs (terminals)
- Submit broadcast messages to groups of terminals represented by an associated broadcast ID

Note: Mailbox credentials are obtained from your IsatData Pro service provider, who also provides a means to associate terminals to each mailbox.

2.4.1: Message retrieval and submission

Applications using the Messaging API (version 1) must actively poll for new return messages or status progression of submitted forward messages.

Forward Message Timeout

If a terminal is unable to receive a forward message, the system will eventually mark the message failed undeliverable and close it. The following table shows how long, after the message has been transmitted, it takes the system to close the message if it cannot be delivered.

Terminal Mode	Terminal fully blocked	Terminal intermittently blocked
Regular	Up to 10 minutes	Up to 60 minutes
Low power	Up to 10 minutes from the scheduled send time	Up to 60 minutes from the scheduled send time

Polling frequency limitations

Inmarsat limits the polling of mailboxes to ensure a balanced system load and fair access policy. Each Mailbox has restrictions on the number of polling operations that will be accepted in a given time interval. API operations that are Rate Limited are indicated in the documentation .

Inmarsat’s *Service Level Agreement* for IsatData Pro specifies the current polling limit for mailboxes - you can obtain this information from your IDP service provider.

Note: Inmarsat may adjust the rate limit from time to time to ensure adequate system performance. Rate limits apply to all applications accessing a given mailbox, so care must be taken when designing multiple applications sharing access to the same mailbox.

2.4.2: Message Retention

Inmarsat’s IDP Message Gateway System retains an active history of messages for a short period of time, to allow Customer Applications to recover from connectivity or other issues. Customer Applications should store messages in their own highly available storage with appropriate backup protection.

Inmarsat’s *Service Level Agreement* for IsatData Pro specifies the current retention period for messages - you can obtain this information from your IDP service provider.

Note: Inmarsat may adjust the retention policy from time to time to ensure adequate system performance. Your IDP service provider should inform you of any such policy changes.

2.5: High watermark

The Message Gateway System uses a 'high watermark' concept to return the latest messages or statuses to customer applications. The high watermark can be represented by either a timestamp or a message sequence number. Along with the latest messages, the MGS returns the next high watermark in both time and message sequence number.

Multiple applications can access the same Mailbox on the MGS. Each application must maintain its own 'high watermark' in order to extract messages in sequence. The customer must save this high watermark and use it for its next access.

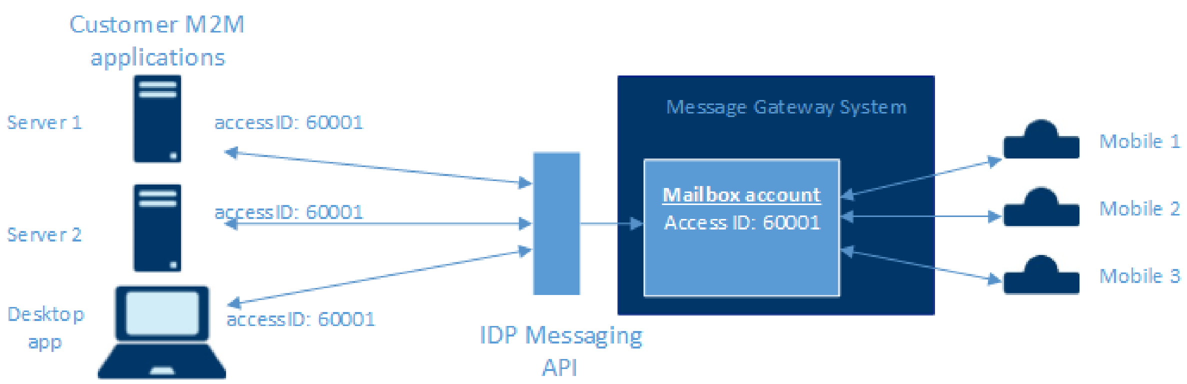

Figure 3. Multiple Applications

Note: Mailbox credentials and polling limitations are effectively shared when multiple applications are accessing the same mailbox. Ensure that customer applications are designed with this in consideration.

2.6: Message Definition Files

Message definitions are an optional feature provided by the IDP Message Gateway System to encode/decode efficient over-the-air (OTA) binary message payloads into tagged data structures for ease of parsing by the Customer Application. The raw OTA binary payload is also retrievable or submittable as an option.

Core modem messages are automatically presented as tagged data structures. Message definitions for messages with SIN 0-15 are reserved and cannot be overwritten by Customer Applications.

The *IDP Message Definition File* specification describes the XML structure that can be provisioned to a mailbox to define your message data structures using a Common Message Framework defined in the specification. Methods (API and/or Portal) to provision a message definition to your mailbox are provided by your IDP service provider.

Note: A mailbox can only support a single Message Definition File, which implies that all terminals associated with that mailbox conform to those message data structures.

2.7: Mobiles (Terminals)

A Mobile must be activated and associated with a mailbox in order to send and receive messages. Your IDP service provider will provide methods to activate or deactivate terminals and associate with mailboxes, based on unique Mobile ID.

Note: Mailbox/mobile associations are one-to-many. While a given mobile ID can only be associated with a single mailbox, a mailbox can support an unlimited number of mobile IDs.

2.7.1: Low Power Mode

Certain IDP terminals may support the system’s low power features, which schedule forward message delivery at a predefined terminal wakeup time. The wakeup interval can be remotely provisioned using Core Modem messages. The specific time that a terminal wakes up is determined automatically by the network, and ensures that wakeup times are evenly distributed to avoid network congestion.

Note: Low power mode only affects forward message delivery. Return messages can be sent by the terminal independent of a low power wakeup interval.

2.8: Billing Considerations

The billing of the service and the messages sent to and from the terminal may depend on the offering from your IDP service provider. In principle, all messages to/from a terminal are billed to the owner of the mailbox to which the terminal is associated.

3: The IDP Messaging API Protocol

3.1: Concept of Operation

Figure 4 describes the basic usage of the Messaging API. A typical client-server application or microservice would be provisioned with all relevant mailbox and mobile/broadcast ID information when initialized, then would periodically:

1. Poll for new return messages (based on most recent high watermark) on a regular interval;
2. On demand, submit forward message(s) to specific mobile or broadcast IDs;
3. After submission, poll forward statuses regularly until completion.

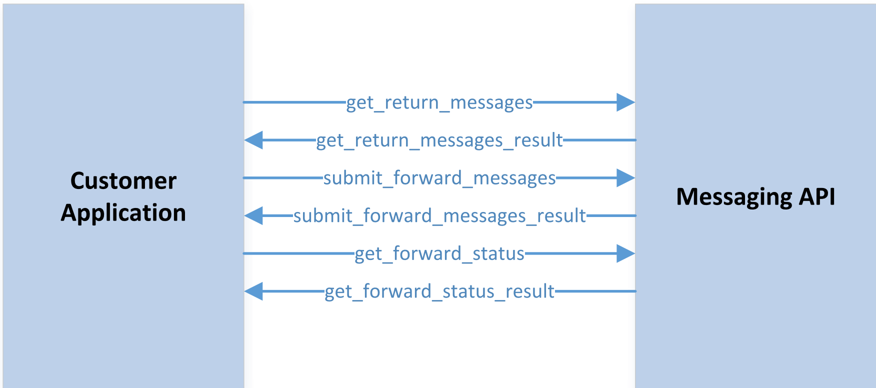

Figure 4. Overview of the IDP Messaging API

3.2: Getting Messages from Terminals

Most IsatData Pro applications are heavily weighted toward periodic and event (report by exception) messages from remote terminals, so the most common operation on the Messaging API is polling to `get_return_messages`.

To start polling, a Customer Application would typically use the current UTC time or a time 12-24 hours in past to establish its initial first high watermark before submitting its first `get_return_messages` operation. The high watermark then gets updated when the API returns a valid `NextStartUTC` which is passed as the `start_utc` filter in the subsequent `get_return_messages` call. Updating the time watermark in this way also allows for recovery of temporary customer application, Internet or API outages.

The Customer Application can selectively retrieve messages based on message ID or timestamp range, particularly if you want to query messages from a specific timeframe in the past. Filtering message retrieval based on a specific mobile ID is also available.

If a Message Definition File has been provisioned on the mailbox, a tagged JSON **Payload** data structure will be returned in the response. Alternatively, an array of byte values (including SIN and MIN) is returned as **RawPayload**.

3.3: Submitting Messages to Terminals

The most common method to send a message to a terminal is using the `submit_messages` operation. A successful submission will generate a response with a unique `ForwardMessageID` that should be stored by the Customer Application in order to track message status to completion.

If a Message Definition File has been provisioned on the mailbox, the Customer Application can submit a JSON Payload data structure. Alternatively, the Customer Application must submit an array of byte values (including SIN and MIN) as **RawPayload**.

The Customer Application may optionally assign a `UserMessageID` during submission, to correlate to the system-generated unique message ID.

Inmarsat limits the number of outstanding to-mobile messages for each terminal. If there are more than 20 outstanding to-mobile messages (consisting of a maximum of 10 messages up to 5000 bytes each, and 10 messages larger than 5000 bytes each) for a single terminal, the API rejects new message submissions and reports an error message.

When a message is submitted it is considered open and remains so until either an acknowledgement is received from the terminal or an error condition occurs. *Figure 5* shows submit message states for a to-mobile message.

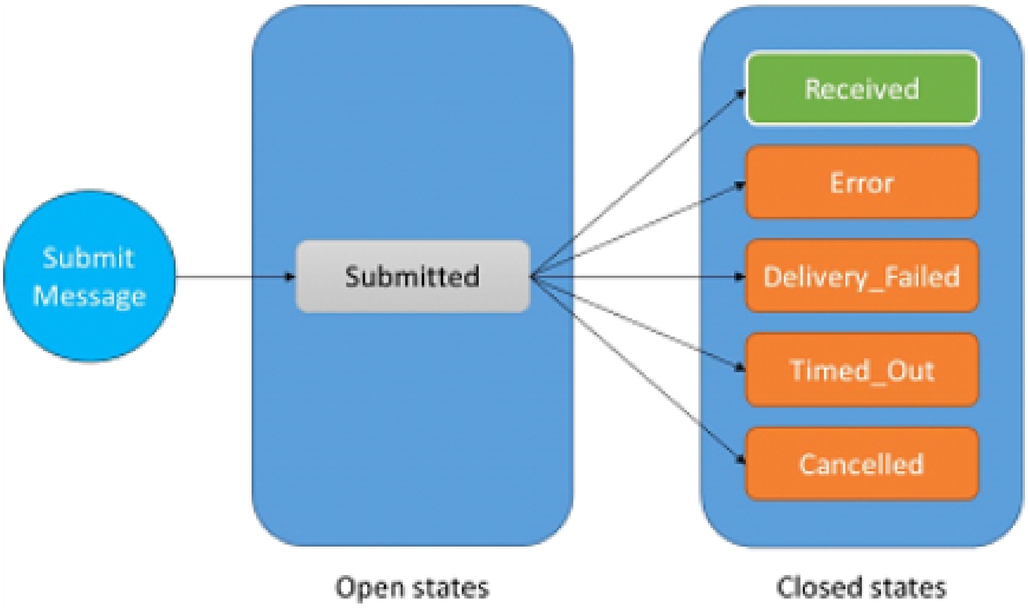

Figure 5. Message States

A message has one of the following states:

Code	State	Description
0	SUBMITTED	Open. The system has accepted the message.
1	RECEIVED	Closed. The system received an acknowledgement from the terminal.
2	ERROR	Closed. An error code specifies the reason.
3	DELIVERY FAILED	Closed. The system failed to deliver the message to a terminal. An error code specifies the reason.

Code	State	Description
4	TIMED OUT	Closed. Message failed to be delivered within 60 minutes. A normal state when the message cannot be delivered is DELIVERY FAILED. TIMED OUT. This state is required in case the message is still not delivered 60 minutes after it was sent from the MGS.
5	CANCELLED	Closed. Message was successfully cancelled by the client application.
6	WAITING	Open. Message is queued. The terminal is in low-power mode and message will be forwarded when terminal wakes up.
7	NA	Not available - message is not recognized.

3.3.1: Submitting Messages to Terminals in Low Power Mode

Messages submitted to terminals using the network low power mode feature are not transmitted immediately. Low power messages are scheduled for delivery during the terminal's next wakeup time. In this case the response to the submit operation will contain a `TerminalWakeupPeriod` and `ScheduledSendUTC` indicating the scheduled delivery time.

Glossary of Terms

Term	Definition
Access ID	An access credential paired with a password, associated with a single Mailbox, to authenticate to the IDP Messaging API.
Broadcast ID	A unique “virtual terminal” that an application can send a Forward message to that will deliver the same message to all the terminals subscribed to that ID.
LES	Land Earth Station is the physical interface to the satellite network, between the MGS and remote terminals.
Mailbox ID	Unique identifier of a Mailbox
MGS	Message Gateway System
Mobile ID	Unique identifier of a remote modem/terminal.