Standard Interchange Protocol 3.0
Part II – Message Structure & Transport

78-8129-4832-7

Contents

Introduction 3
Message Structure & Rules 3
Character Encoding 3
Field Terminator 3
Message Terminator 3
Nulls 3
Message Identifiers 4
Message Syntax 4
Fields 4
Data sets 5
Transport Methods 7
Packet Format 7
Data Encryption 7
Behavioral Rules 8
Messages 8
Fields 8
Data Sets 8
Display and Print messages 8
Version Validation 8

3M™ Standard Interchange Protocol

Part II – Message Structure & Transport

Introduction

This document defines a standard message structure and transport mechanisms to allow interoperability between vendor products utilizing the SIP 3.0 protocol.

The SIP 3.0 protocol specification has been divided into two documents. Part I defines the commands, message content and field definitions. Part II defines how to structure the messages and how to transmit those messages to allow interoperability. As technology changes around us there may be a need or desire to change the way these messages are structured or how they are transmitted. This will allow new transport mechanisms to be created and implemented without changing Part I of the protocol.

Message Structure & Rules

This section defines the formatting of all SIP messages.

Character Encoding

The entire message shall be encoded using UTF-8 character encoding standard.

Field Terminator

All fields must end with a field terminator. The field terminator is a hexadecimal 7c. This character cannot be used elsewhere in the message.

Message Terminator

All messages must end in a carriage return (hexadecimal 0d). This character is interpreted as the last character in a message and cannot be used elsewhere as a character in a message.

Nulls

Null codes (hexadecimal 00) cannot appear anywhere in a message.

Message Identifiers

Message Identifiers (Message Ids) are two characters that uniquely identify the message. These are defined in Part I of the protocol specification. The message id must be the first characters in the message being transmitted.

Message Syntax

Messages consist of a message identifier, fields, and a message terminator. The message identifier must be the first two characters of the message followed by one or more fields, each terminated with a field terminator, and then ending with a message terminator.

<Message Identifier><fields><Message terminator>

Examples:

Fields

Fields may be sent in any order. Mandatory fields must appear in the message. Optional fields may or may not appear in the message. It is recommended that the AY field be the last field in the message to make debugging easier for developers.

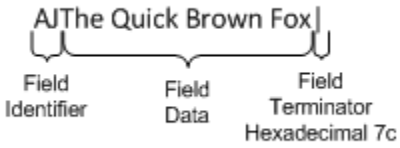
A field may be repeated if multiple occurrences are allowed.

For some fields where a group of fields may be repeated, such as list data and order must be maintained, the protocol specifies the required field order.

Field Syntax:

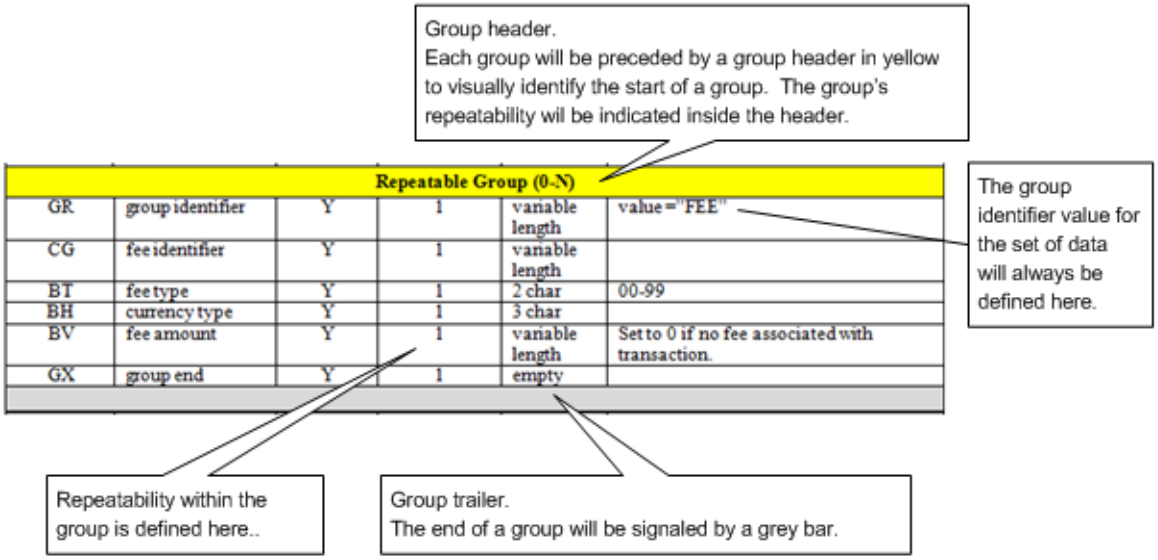
<Field Identifier><Field data><Field terminator>

Example:

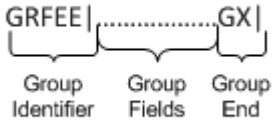


Data sets

A group represents a set of data that can appear in a message structure. A group will appear as in the protocol as follows.



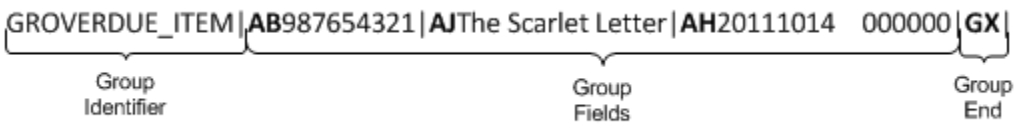
Group data must always start with a group identifier (GR) followed by one or more fields and end with a group end field (GX).



The Group identifier (GR) field must indicate the type of data that is defined by the group. Valid values are defined in the protocol Part I document. The type defines mandatory and optional fields within the group and the repeatability of each field. If the GR field does not contain the type of group in the field, the group is unknown and should be ignored, similar to the way an unknown field should be handled.

Group examples:

Overdue Item group



Fee group

Transport Methods

SIP messages can be transmitted using TCP/IP and telnet in secure environments. For sites that require the data to be encrypted we recommend the use of SSH or TLS protocols. (TLS is the successor to SSL; see http:/en.wikipedia.org/wiki/Transport Layer Security.)

Packet Format

Messages to and from the ILS have the same general format. The message packet begins with a message identifier. The message identifier is followed by fields with field identifiers, each of which is terminated with a field terminator. The message ends with a carriage return.

Data Encryption

Direct TCP/IP socket and telnet communications between the self-service system and the ILS should only be used where there is a dedicated network for the self-service systems. Where the network is shared between other applications or users, communications via TLS (Transport Layer Security) or SSH (Secure Shell) is preferred to ensure that the entire transaction between the self-service system and the ILS is encrypted.

TLS would be a direct replacement where a self-service system communicates to the ILS via TCP/IP sockets. TLS is implemented directly on TCP, and many TCP/IP socket implementations will allow the creation of TLS enabled socket communication using similar APIs to unsecured sockets. During the initial socket connection, an additional handshake is performed to determine the encryption algorithms used. During this handshake the ILS would identify itself via a PKI certificate. Optionally, the self-service system could use this certificate to confirm that it is communicating to the correct ILS. Where an ILS supports both unencrypted sockets and TLS sockets, these would be available on different TCP ports on the ILS. There is no mechanism in the SIP protocol to negotiate between unencrypted and TLS sockets on the same TCP port.

SSH would be a direct replacement where a self-service system communicates to the ILS via the telnet protocol. There are programming libraries for SSH which would provide a simple replacement for telnet connections. In terms of sending messages between the self-service system and the ILS, messages would be sent via SSH connection in a similar manner to sending messages via a telnet connection. The main differences would occur when making the initial SSH connection, in that:

1. SSH performs a key exchange at the initial connection, and after a set amount of data has been transferred to establish the encrypted connection. Since the ILS uses a PKI certificate, it is possible for the self-service system to use this certificate to confirm that it is communicating to the correct ILS.

2. SSH has its own authentication mechanism rather than sending a login prompt. The ILS could be configured to always accept incoming connections or could use the SSH authentication mechanisms (which include username and password as well as client certificate) in addition to or instead of the SIP protocol login message.

Behavioral Rules

Messages

Message identifiers that are unrecognized should be ignored or return an Unsupported Message Response. This allows new commands to be added to the protocol in the future without adversely affecting software written for earlier versions of the protocol.

All recognized SIP commands sent by the self-service system to the ILS require a response from the ILS.

Fields

Fields with unrecognized field identifiers shall be ignored. This allows new fields to be added to the protocol in the future without adversely affecting software written for earlier versions of the protocol.

When optional fields are not used, they should be left out entirely.

Data Sets

Groups with an unrecognized group type shall be ignored. This allows new groups to be added to the protocol in the future without adversely affecting software written for earlier versions of the protocol.

Display and Print messages

Only displayable characters (no control characters) should be included in print or display messages from the ILS.

Version Validation

If the Login Message is required, it must be the first message exchanged, and the version data must be sent with the message. If the Login is not required the first message communicated will be the SS Status message, and that message will contain the version information.