

i-Page Protocol (iPP)

Version 1.1 and 1.2



WiPath Communications Ltd

Contents at a Glance

Introduction.....	6
Message Format.....	8
Message Size.....	8
Sending Message.....	9
Receiving Message.....	9
Message Header.....	10
Main Message.....	18
Messages.....	21
Object Fields.....	53
Appendix A.....	85
Appendix B.....	97
Appendix C.....	103
Appendix D.....	107
Appendix E.....	110

Table of Contents

Introduction.....	6
i-Page.....	6
i-Page Server.....	6
i-Page Client.....	6
Protocol.....	6
Message Format.....	8
Message Size.....	8
Message Size Format.....	8
Sending Message.....	9
Receiving Message.....	9
Message Header.....	10
Header Format.....	10
Fixed part.....	10
Variable Part.....	11
Message Header Fields.....	11
Protocol Version.....	12
Command Type.....	12
Command Subtype.....	13
Message Id.....	14
Field Delimiter.....	15
Date Received.....	15
Date Returned.....	16
Client Data.....	16
Header Delimiter.....	17
Main Message.....	18
Main Message Defined.....	18
Messages.....	21
Message Sequence.....	21
Connect And Login.....	24
Connect.....	24
Login Request.....	24
Login.....	25
Connection Accepted.....	26
Login Success.....	26
Client Ready.....	26
Connection Status.....	27
Client Not Connected.....	27
Server Disconnects.....	27
Administrator Disconnects.....	27
Client Disconnects.....	27
Connection and Login Errors.....	29
Connection Excess.....	29
Account Disabled.....	29
Login Error.....	29
Duplicate Login.....	29
Send Message.....	31
Send Message.....	31
Send Message Result.....	34
Loading Objects.....	37
Load Request.....	37
Load Success.....	38
Load Error.....	39
Manipulating Message Objects.....	41
New Object.....	41
Save Object.....	43
Delete Object.....	43

Find Object.....	44
Assign Objects.....	46
Other Messages.....	48
Change Password.....	48
Change Account Settings.....	49
Run Report.....	50
Query Sent Messages.....	51
Object Fields.....	53
System Objects.....	53
Account.....	53
System.....	56
Carrier.....	58
Message Settings.....	60
Sent Message.....	64
Message Objects.....	67
Contact.....	69
Group.....	70
Template.....	71
Schedule.....	72
Report.....	76
Folder.....	83
Appendix A.....	85
List Of Commands.....	85
List Of Command Subtypes.....	87
Connection Subtypes.....	87
Login Subtypes.....	89
System Object Subtypes.....	90
Send Message Subtypes.....	94
List Of Control Characters.....	96
Appendix B.....	97
Constants.....	97
Message Priorities.....	97
Message Affix Types.....	97
Address Type.....	97
Carrier Type.....	98
Text Encoding Type.....	98
Sent Messages Query Type.....	98
Find Object Query Type.....	98
Contact Display Type.....	99
Tree View Display Type.....	99
Message Delete Type.....	99
Schedule Type.....	100
Schedule End Type.....	100
Report Date Range Type.....	100
Stored Messages Fields.....	100
Message Sent Status.....	101
Text Search Type.....	101
Auto Report type.....	102
Appendix C.....	103
Data Types.....	103
String.....	103
Boolean.....	104
Number.....	104
Integer.....	104
DateTime.....	104
IP Address.....	105
Appendix D.....	107

TCP/IP Byte Order.....	107
Host To Network.....	107
Network To Host.....	107
Delimiting Strings.....	109
Appendix E.....	110
Message Sequence Example.....	110

Introduction

i-Page

Message Dispatch Software for Paging and Text Messaging

i-Page is a communications software for the dispatching of text or SMS messages to a variety of wireless messaging devices. It is primarily designed for networked client/server operations.

i-Page utilizes TCP/IP and COM communications between the server and clients, enabling clients on a local and/or wide area network to send messages to pagers and other text capable wireless devices such as cellular phones, MDTs, etc.

i-Page Server

The i-Page Server software is located on a centralized PC with access to the carriers to which it is to communicate. The server may have any number of carriers of various protocols and communications mediums and which may be local or remote (see Communications protocols).

Every user must have an account created on the server to be able to access the server. Levels of access are controlled by the users' rights, set by the system administrator.

Once they have an account, every user can create their own list of contacts and message templates. They can also create their own contact groups to allow them to send a single message to multiple contacts in one hit. The group can consist of contacts of different types connected to different carriers.

The system administrator can create contacts, contact groups and message templates that are visible to all accounts. The user can use those objects in the same way as their own, but they cannot edit or delete them.

All contacts, contact groups and message templates are stored on the server.

The server provides COM, TCP/IP file and web connection to enable clients to communicate with it. The user on the client can select a connection type suitable to them.

i-Page Client

i-Page Client is loaded on to each PC that requires the ability to send messages. Clients may be located anywhere on the LAN or WAN with normal access to the server. To access the server the client must complete the log on procedure. If the authentication is successful the server sends to the client all application and account settings and all the message objects (contacts, contact groups, message templates, schedules, reports) that are associated with that account.

Protocol

i-Page Client/Server Protocol (**iPP**) defines a network communication between the i-Page Server and any type of i-Page client application.

It is an application level protocol that runs over TCP/IP and defines the functions to be performed by the protocol and the program that implements it.

iPP uses TCP as its transport protocol and IP as its network (routing) protocol.

Protocol Stack

Application Layer	iPP
Transport Layer	TCP
Network Layer	IP
Data-link Layer	device driver and interface card

Message Format

Every message used by i-Page Client/Server Protocol has the following format

Message Size	Message Header	Header Delimiter	Main Message
2(4) bytes	Variable number of bytes	1 byte	Variable number of bytes

Example:

\$6C00↵

```
11 CONNECT0C000020D9,7DD6060C0D2D1F76BC1E0000001Dnicole2Client "nicole" on
10.58.2.199 ready to communicate.
```

- The message above uses “Normal” size format. It contains 108 (\$6C) bytes and the size is expressed in [TCP/IP network byte order](#) (27,648 - \$6C00).

If the message uses “Large” size format, the same size expressed in [TCP/IP network byte order](#) would be \$6C000000 (1,811,939,328).

Message Size

The message size defines the **number of bytes** that the whole message consists of. It includes:

- Message header size
- Header delimiter size
- Main message size

It **does not include** 2(4) bytes for the **message size field**.

Message Size Format

Every message is preceded with a number that specifies the number of bytes that the message consists of and delimit its size.

i-Page Server supports two types of message sizes:

- Normal** (default)
 - The size is expressed as a **signed word** (16-bit signed number) in [TCP/IP network byte order](#)
Max Value: 32,767 bytes
- Large**
 - The size is expressed as a **signed double word** (32-bit signed number) in [TCP/IP network byte order](#)
Max Value: 2,147,483,647 bytes

On Intel platforms it is usually necessary to convert from host byte order (which is little-endian on Intel processors) to TCP/IP network byte order (which is big-endian) and other way around.

Sending Message

To send the message:

1. Create the message
2. Convert the whole message into the UTF-8 encoding form.
3. Count the number of bytes that the message contains
4. Convert the number of bytes into network byte order
5. Write the number of bytes in network byte order to the socket
6. Write the whole message to the socket

Receiving Message

To receive the message:

1. Read the first 2(4) bytes of the message as the number of the message bytes
2. Convert the the number of the message bytes to host byte order
3. Read that number of bytes (the whole message) from the socket.
4. Convert the message from the UTF-8 encoding form to whatever encoding the client is using.
5. Decode the message according to this protocol.

Message Header

Header Format

Every message used by i-Page Protocol has the same header format. It consists of a fixed size part and a variable size part.

Header Field	Field Size	Field Position
Fixed Part		
Protocol	2	1
Command	10	3
Command Subtype	2	13
Message Id	8	15
Field Delimiter (FD)	1	23
Variable Part		
Date Received	(can be empty) 12	24
Dates Delimiter (US)	1	Variable
Date Returned	(can be empty) 12	Variable
Client Data Delimiter (GS)	1	Variable
Client Data	(CANNOT be empty) Variable	Variable
Header Delimiter (STX)	1	Variable
Header Data Length	28 +	

All field sizes are expressed in bytes.

The field position shows the position of the field data in a message byte array, starting with 1.

Fixed part

- 23 bytes
- Character format: [ASCII String](#)
- Fields are byte counted
- Contains data that are common to all messages:
 - protocol version
 - message command

- command subtype
- message id
- field delimiter

Variable Part

- Variable size
 - “Date Received” and “Date Sent” fields can be empty
 - “Client Data” field can have a variable size, but it cannot be empty
- Contains
 - received date
 - date delimiter
 - sent date
 - client data delimiter
 - client data
 - header delimiter
- Character format:
 - “Client Data” field - [Unicode String](#)
 - All other fields - [ASCII String](#)
- Positions:
 - The variable part starts immediately after the fixed part, at byte position 24
 - The whole part is delimited by the “Header Delimiter” character.
 - The “Date Received” field
 - ◇ Starts at byte position 24
 - ◇ Delimited with the date delimiter ([US](#) - [\\$1F](#)) character
 - The “Date Sent” field
 - ◇ Starts at the date delimiter ([US](#) - [\\$1F](#)) character
 - ◇ Delimited with the client data delimiter ([GS](#) - [\\$1D](#)) character
 - The “Client Data” field
 - ◇ Starts at the client data delimiter ([GS](#) - [\\$1D](#)) character
 - ◇ Delimited with the header delimiter ([STX](#) - [\\$2](#)) character

Message Header Fields

To be considered valid, the header must have at least 28 bytes. It must contain:

Fixed part	24 bytes
Dates delimiter	1 byte
Client data delimiter	1 byte
Client data	1 byte minimum
Message delimiter	1 byte

The header size is increased by the “Date Received”, “Date Returned” and “Client Data” field sizes.

Every field in the header, except “Client Data”, can contain only ASCII characters. The “Client Data” field can contain any Unicode character.

Protocol Version

Ensures future and backward compatibility between different versions of iPP protocols.

- Size: 2 bytes – fixed size length.
- Position: 1st byte
- Message Format:
 - 2 hexadecimal characters
 - First character indicates a major protocol version
 - Second character indicates a minor protocol version.
- Usage:
 1. The version of the protocol is negotiated between a client and the server for every session. The negotiation takes place before the session is established.
 2. In its first message - “Connect” (**CONNECT CR_CONNECT**) - the client sends the highest version of the protocol that it can handle.
 3. The server compares the client's protocol version with the highest version it understands and calculates the highest version that both, the client and the server can handle.
 4. In its first message - “Login Request” (**LOGIN LR_LOGIN_REQUEST**) - the server returns a protocol version's common denominator.
 5. The client remembers this common version's value for the whole session and uses it in each subsequent message.
- Position and size of this field cannot be changed in any future protocol version.

Command Type

Defines the type of the message.

- Size: 10 bytes – fixed size length.

- Position: 3rd byte
- Message Format:
 - [ASCII String](#), defined by the protocol – See [List Of Commands](#)
 - ◇ If a command string is less than 10 characters long, it is left padded with ASCII space character ([\\$20](#)).
 - ◇ Not case sensitive, but i-Page Server always returns a command in upper case
- Usage:
 - The software that receives the message, reads the “[Command](#)” field and the “[Command Subtype](#)” field and according to their values:
 - ◇ Decodes the other message fields
 - ◇ Takes an appropriate action

Examples:

'...CONNECT' – connect command

'.....LOGIN' – login command

'.....LOAD' – load command

'...OBJ_NEW' – create a new message object command

'...QUERY_RUN' – find a system-wide message object command

'...MSG_SEND' – send message command

Command Subtype

Further specifies the message command. It advises the receiving side how to decode the rest of the message, what actions to take or what was the result of the server's action.

- Size: 2 bytes – fixed size length
- Position: 13th byte
- Message Format:
 - Uses different sets of constants for different message commands. See [List Of Command Subtypes](#)
 - All constant values are in 1 byte number range.
 - The constant is expressed as 2 hexadecimal characters
 - If a result string is less than 2 hex characters long, it is left padded with '0' (zero) character ([\\$30](#)).
 - Not case sensitive, but i-Page Server always returns hexadecimal characters in upper case.
- Usage:

- The software that receives the message, reads the “[Command](#)” field and the “[Command Subtype](#)” field and according to their values:
 - ◇ Decodes the other message fields
 - ◇ Takes an appropriate action

Examples:

- '...CONNECT01' - The client requests a connection to the server
- '...CONNECT06' - The server has accepted the client's connection
- '...CONNECT07' - The server has refused the client's connection, because it exceeds the number of registered connections
- '...CONNECT07' - The server is closing down
- '.....LOGIN01' - The server is requesting login
- '.....LOGIN00' - The client is sending its login information to the server
- '.....LOGIN03' - Login Error. Incorrect user name and/or password
- '...MSG_SEND00' - The client is sending a page, SMS or email message to the server.
- '...MSG_SEND01' - The message is sent to the carrier.
- '...MSG_SEND14' - A carrier has reported an error while trying to send the message.

Message Id

A number that uniquely identifies the message to the sender. It is assigned by the side that has originated the message (client or server). If the client does not want to follow up the message result it can set this field to '0'

- Size: 8 bytes – fixed size length
- Position: 15th byte
- Message Format: [Integer](#)
- Usage:
 - The main purpose of this field is to be used with the “[Send Message](#)” command.
 - ◇ When the client sends a page, SMS or email message to the server, it has to wait for an undefined amount of time for the result of the send operation. This happens due to the asynchronous nature of the TCP connection and due to the time that the server needs to process the message and send it to the carrier.
 - ◇ The client can assign a unique transaction identifier to every message it sends.
 - ◇ When the server returns the result of the send message operation, it also returns the same transaction id that was assigned by the client.
 - ◇ The client can now match the original message with its result.
 - With all the result [command subtypes](#) the server returns the transaction id that is

assigned by the client in its request.

Field Delimiter

Used as a delimiter between different data elements in the “[Main Message](#)” field.

- Size: 1 byte – fixed size length.
- Position: 23rd byte
- Message Format: Any ASCII character can be used, except [RS \(\\$1E\)](#) character.
- Usage:
 - It is assigned by the sender
 - ◇ Default: **','** – ASCII **'COMMA'** character ([\\$2C](#)).
 - ◇ The client and the server use it to delimit different parts of data in the “Main Message” field.
 - The receiver uses it to parse the same fields into tokens.
- Care must be taken if delimited strings already contain the “[Field Delimiter](#)” character. See [Delimiting Strings](#)

Date Received

Date and time when the server has received the message from the client.

- Size: 12 bytes (if defined)
- Position:
 - Starts after the [Field Delimiter](#) character (end of the fixed part – 24th byte)
 - Ends with the date delimiter character ([US - \\$1F](#))
- Message Format: [DateTime](#)
- Usage:
 - The value of this field is assigned by the server, when it receives the message
 - When sending the message, the client can assign some value to this field. The value can be:
 - ◇ Empty string ("")
 - ◇ One or more spaces ('.....')
 - ◇ One or more zeroes ('0000')
 - ◇ If some other value is entered, it has to be a valid [DateTime](#) value

- The server always returns a valid [DateTime](#) value

Date Returned

Date and time when the server has returned a message result or an error to the client.

- Size: 12 bytes (if defined)
- Position:
 - Starts with the date delimiter character ([US](#) - [\\$1F](#))
 - Ends with the client data delimiter character ([GS](#) - [\\$1D](#))
- Message Format: [DateTime](#)
- Usage:
 - The value of this field is assigned by the server, when it receives the message
 - When sending the message, the client can assign some value to this field. The value can be:
 - ◇ Empty string ("")
 - ◇ One or more spaces (' ')
 - ◇ One or more zeroes ('0000')
 - ◇ If some other value is entered, it has to be a valid [DateTime](#) value
 - The server always returns a valid [DateTime](#) value

Client Data

Contains the id of the account that is logged on the server. This field is set by the client and cannot be changed by the server. The server always returns the same value to the client.

- Size: variable size
- Position:
 - Starts with the data delimiter character ([GS](#) - [\\$1D](#))
 - Ends with the “[Header Delimiter](#)” character ([STX](#) - [\\$2](#))
- Message Format: [Unicode String](#)
- Usage:
 - Contains the account id of the user that is logged on
 - The id must be registered on the server
 - The field is set by the client
 - The server always returns the same value to the client

- It can be assigned only with the [Login](#) command.
- Before that, it can contain any [Unicode String](#) that is not used as one of the account ids registered on the server
 - ◇ Default – **'Unknown'**
- It **CANNOT** be an **empty** string.

Header Delimiter

The header delimiter is an ASCII character that separates the message header from the main message field.

Size - 1 byte.

In this version [STX \(\\$2\)](#) character is used.

Main Message

The “Main Message” field carries a real payload of the message. It contains data that the client sends to the server or the server's response to the client. Its content depends on:

1. Type of the message – defined by its “[Command Type](#)” field
2. Subtype of the message – defined by its “[Command Subtype](#)” field
3. Direction of the message – client/server or server/client

Its content is defined for some message types. If it is not defined it can be either empty or contain a description of the command. The server never returns an empty “Main Message” field. If not defined for that particular type of the message, the server returns description of the command or its result.

Main Message Defined

Commands and command subtypes for which the “Main Message” field is defined:

Command Type	Command Subtype	Sender	Contains
MSG_SEND	SMR_MSG_SEND	Client	One or more records, where each record consist of: ◇ Paging, SMS or email message send by the user ◇ Message data, separated into fields
LOGIN	LR_LOGIN	Client	Account's login data ◇ user name ◇ password ◇ encoding type
LOGIN	LR_CHANGE_LOGIN	Client	Account's login data ◇ old password ◇ new password ◇ confirmed password ◇ encoding type

Command Type	Command Subtype	Sender	Contains
LOAD	LOR*_OK ◇ LOR_CONTACT_OK ◇ LOR_GROUP_OK ◇ LOR_TEMPLATE_OK ◇ LOR_SCHEDULE_OK ◇ LOR_REPORT_OK ◇ LOR_CARRIER_OK ◇ LOR_ACCOUNT_OK ◇ LOR_SYSTEM_OK ◇ LOR_MSG_SETUP_OK	Server	One or more records, where each record consists of properties for one object, divided into fields
OBJ_NEW	LOR*_OK ◇ LOR_CONTACT_OK ◇ LOR_GROUP_OK ◇ LOR_TEMPLATE_OK ◇ LOR_SCHEDULE_OK ◇ LOR_REPORT_OK	Server	Record with properties of a newly created object, separated into fields
OBJ_SAVE	LOR_* ◇ LOR_CONTACT ◇ LOR_GROUP ◇ LOR_TEMPLATE ◇ LOR_SCHEDULE ◇ LOR_REPORT ◇ LOR_MSG_SETUP	Client	Record with edited properties of the object, separated into fields
OBJ_DEL	LOR_* ◇ LOR_CONTACT ◇ LOR_GROUP ◇ LOR_TEMPLATE ◇ LOR_SCHEDULE ◇ LOR_REPORT	Client	System ID of the object to be deleted
QUERY_RUN	LOR_* ◇ LOR_CONTACT ◇ LOR_GROUP ◇ LOR_TEMPLATE ◇ LOR_SCHEDULE ◇ LOR_REPORT	Client	◇ Search type ◇ Search string

Command Type	Command Subtype	Sender	Contains
QUERY_RUN	LOR_*_OK ◇ LOR_CONTACT_OK ◇ LOR_GROUP_OK ◇ LOR_TEMPLATE_OK ◇ LOR_SCHEDULE_OK ◇ LOR_REPORT_OK	Server	One or more records, where each record consists of properties for one found system-wide object, divided into fields
QUERY_RUN	LOR_RUN_REPORT	Client	System id of the selected report object
QUERY_RUN	LOR_SENT_MSG	Client	◇ Send messages query type ◇ Number of messages to return ◇ Messages sent from date ◇ Messages sent to date
QUERY_RUN	LOR_SENT_MSG_OK	Server	One or more records, where each record consists of properties for one sent message object, divided into fields
ACC_CHNG	LOR_ACCOUNT	Client	Record with edited properties of the account, divided into fields
ACC_CHNG	LOR_* ◇ LOR_CONTACT ◇ LOR_GROUP ◇ LOR_TEMPLATE ◇ LOR_SCHEDULE ◇ LOR_REPORT	Client	System Ids of the objects of the selected type assigned to the account. Ids are separated with the " Field Delimiter " character, defined in the "Header".

For All Other Commands:

- The "[Main Message](#)" field is optional
- It can be generated either by the server or by the client
- It can contain a description of the command, action performed or result of the action
- Format:
 - [Unicode String](#)
 - It is never enclosed in any type of quotes, but it can contain quoted parts.

Messages

Message Sequence

Client	Server	Server – Errors
Requesting connection to the server CONNECT CR_CONNECT (CONNECT01)	Requesting authentication LOGIN LR_LOGIN_REQUEST (LOGIN01)	
Logging on LOGIN LR_LOGIN (LOGIN00)	Successfully connected to the server CONNECT CR_CONNECTION_ACCEPTED (CONNECT05)	
	Successfully logged on the server LOGIN LR_LOGIN_OK (LOGIN02)	Connection refused because the number of clients exceeds the number of allowed connections CONNECT CR_CONNECTION_EXCESS (CONNECT06)
		Login error – user name and/or password incorrect LOGIN LR_LOGIN_ERROR (LOGIN03)
		Login error – account disabled LOGIN LR_ACCOUNT_DISABLED (LOGIN04)
		Login error – duplicate login LOGIN LR_DUPLICATE_LOGIN (LOGIN06)

Client	Server	Server – Errors
[Optional Loading system objects settings] LOAD LOR_* (LOAD0*) (00 - 07)	[Optional Settings for requested objects successfully loaded] LOAD LOR_*_OK (LOAD0*) (08 - 0F)	[Optional Error loading system objects settings] LOAD LOR*_ERROR (LOAD1*) (10 - 17)
Client ready to communicate with the server CONNECT CR_CLIENT_READY (CONNECT0B)		
Send message MSG_SEND SMR_MSG_SEND (MSG_SEND00)	Message successfully sent to the carrier MSG_SEND SMR_MSG_SEND_OK (MSG_SEND01)	
Logging out and disconnecting from the server CONNECT CR_CLIENT_DISCONNECT (CONNECT08)		Send message errors MSG_SEND SMR_* (MSG_SEND**) (02 - 19)

The client can communicate with the server that is running on the same machine, on any machine on the local network or anywhere on the Internet. To communicate with the server, the client must first establish TCP/IP connection through the socket interface. For successful connection following conditions must be met:

1. The client must know the server's IP address

2. The client must know the server's TCP port
3. The machine on which the server is running must be visible to the client
4. The server must be running and started

Connect And Login

Connect

After a TCP/IP connection is established, the very first message that the client sends to the server is the “Connect” (**CONNECT**) command.

Sender	Command	Subtype	Example
Client	CONNECT	CR_CONNECT	...CONNECT01

Example:

```
11...CONNECT01000020B4,7DD6060A35371F76BC1E000001DUnknown
2Connecting to i-Page Server on 10.58.2.199:6022
```

Header

□ Protocol

- This message starts the iPP protocol negotiation between the client and the server
- The client and the server must establish which is the highest version of the protocol they are both able to handle
- The client sends the highest version of the iPP protocol that it knows how to handle

Example:

12

□ Client Data

- In this moment (before the login procedure) It is not known to the client which account is trying to connect to the server
- The account id cannot contain any valid account id.
- It can contain any [Unicode String](#) that is not used as one of the account IDs registered on the server.
Default – **'Unknown'**
- It **cannot** be an **empty** string.

Example:

1DUnknown2

Login Request

The server responds to the “[Connect](#)” command by sending the “Login Request” command

Sender	Command	Subtype	Example
Server	LOGIN	LR_LOGIN_REQUESTLOGIN01

Example:

```
11.....LOGIN01000020DA,7DD6060E2F121F7DD6060E2F121DUnknown
2Connection not accepted by i-Page Server 0D0A The server requests login
```

Header□ **Protocol**

- This message completes the iPP protocol negotiation between the client and the server
- The server compares version of the protocol received from the client in the “[Connect](#)” message with its highest version and finds the highest common version that they are both able to handle
- The server sends the version of the iPP protocol that will be used in the current session
- The client should remember the version of the iPP protocol and use it in all other messages throughout the session.

Example:

```
11
```

Login

The client responds to the “[Login Request](#)” command by obtaining login information from the user and sending them as a part of the “Login” (**LOGIN**) command.

Sender	Command	Subtype	Example
Client	LOGIN	LR_LOGINLOGIN00

Example:

```
11.....LOGIN00000020C2,7DD6060B0E251F76BC1E0000001Dnicole2nicole,nicky,0
11.....LOGIN00000020B5,7DD6060B0D141F76BC1E0000001Dnicole
2nicole,7F7141132F905A74836EA2C53711904F,1
```

Main Message

- Contains 3 fields with the account's login information:
- Fields are delimited with the “[Field Delimiter](#)”, defined in the header.
- Fields:
 - User Name
 - ◇ The account's ID, as registered on the server
 - Password
 - [Encoding type](#)
- The “User Name” and “Password” fields must be present.
- The “Password” field can be encoded or in plain text.

- The encoding type field can be empty or missing. In that case the server will consider the password to be a plain text.
- To perform encoding, the client needs to use “*IPCrypt.dll*” library.

Example:

```
2nicole,7F7141132F905A74836EA2C53711904F,1
2nicole,nicky,0
2nicole,nicky,·
2nicole,nicky
```

Connection Accepted

The server responds to the “Login” command by sending several status messages to the client. The first one is the result of the “Connect” command.

Sender	Command	Subtype	Example
Server	CONNECT	CR_CONNECTION_ACCEPTED	...CONNECT05

Example:

```
11...CONNECT05000020DC,7DD6060E34041F7DD6060E34041Dnicole
2Connection accepted by i-Page Server 0D0A Connection no 3 of 10
```

Login Success

The server responds to the “Login” command by sending several status messages to the client. The second one is the result of the “Login” command.

Sender	Command	Subtype	Example
Server	LOGIN	LR_LOGIN_OKLOGIN02

Example:

```
11.....LOGIN02000020DC,7DD6060E34051F7DD6060E34051Dnicole
2Account "nicole" successfully logged on i-Page Server
```

Client Ready

If the client connects successfully to the server and passes the authentication procedure, the server waits for the client's message that it is ready to communicate. In the meantime the client may want to download some of the server settings – any of the server objects or message objects.

Sender	Command	Subtype	Example
Client	CONNECT	CR_CLIENT_READY	...CONNECT0B

Example:

```
11...CONNECT0C000020D9,7DD6060C0D2D1F76BC1E0000001Dnicole
2Client "nicole" on 10.58.2.199 ready to communicate.
```

Connection Status

The server and the client can send several different status message. All of them define the state of the client – server connection.

Client Not Connected

The server sends this message to the client to advise it that it is not connected to the server.

Sender	Command	Subtype	Example
Server	CONNECT	CR_NOT_CONNECTED	...CONNECT0B

Example:

```
11...CONNECT0B000020DC,7DD6060E34041F7DD6060E34041Dnicole
2Not connected to i-Page Server on 10.58.2.199:6022
```

Server Disconnects

The server sends this message to the client to advise it that it is disconnecting form this client because the server is closing down.

Sender	Command	Subtype	Example
Server	CONNECT	CR_SERVER_DISCONNECT	...CONNECT07

Example:

```
11...CONNECT0700003E44,7DD6060F0C251F76BC1E0000001Dnicole
2i-Page Server on 10.58.2.199:6022 is closing down
```

Administrator Disconnects

The server sends this message to the client to advise it that the server administrator has disconnected the client.

Sender	Command	Subtype	Example
Server	CONNECT	CR_ADMIN_DISCONNECT	...CONNECT09

Example:

```
11...CONNECT0900000000,7DD6060F091E1F7DD6060F091E1Dnicole
2Your connection to i-Page Server has been disconnected by the server
administrator
```

Client Disconnects

The client sends this message to the server to advise it that it is logging out and disconnecting form the server.

Sender	Command	Subtype	Example
Client	CONNECT	CR_CLIENT_DISCONNECT	...CONNECT08

Example:

```
11...CONNECT080000234F,7DD6190D12161F76BC1E000001Dnicole2Account "nicole"  
logging out from i-Page Server'
```

Connection and Login Errors

Connection Excess

The server sends this message to inform the client that its connection request has been refused because the number of concurrent client connections has already been reached.

Sender	Command	Subtype	Example
Server	CONNECT	CR_CONNECTION_EXCESS	...CONNECT06

Example:

```
11...CONNECT060000237E,7DD6190D2D071F7DD6190D2D071Dnicole2Connection refused by
i-Page Server 0D0A The number of clients that the server is registered for (12)
exceeded'
```

Account Disabled

The server sends this message to inform the client that the authentication has failed because the account that tries to log on is disabled by the system administrator.

Sender	Command	Subtype	Example
Server	LOGIN	LR_ACCOUNT_DISABLEDLOGIN04

Example:

```
11.....LOGIN0400002351,7DD6190D1B2C1F7DD6190D1B2C1Dnicole2i-Page Server does not
accept your login 0D0A The account "nicole" is deactivated by the administrator
```

Login Error

The server sends this message to inform the client that the authentication has failed because either the user name or password (or both) is incorrect.

Sender	Command	Subtype	Example
Server	LOGIN	LR_LOGIN_ERRORLOGIN03

Example:

```
11.....LOGIN0300002353,7DD6190D20051F7DD6190D20051Dnicoles2i-Page Server does
not accept your login 0D0A Incorrect user name and/or password
```

Duplicate Login

The server sends this message to inform the client that the authentication has failed because the same account from the same machine is already logged on.

Sender	Command	Subtype	Example
Server	LOGIN	LR_DUPLICATE_LOGINLOGIN06

Example:

11.....LOGIN060000237C,7DD6190D27381F7DD6190D27381Dnicole2i-Page Server does not accept your login 0D0A The account "nicole" on 10.58.2.199 is already logged on.

Send Message

Send Message

After successful connection and authentication, the client can send a user's page, SMS or email message to the server.

Sender	Command	Subtype	Example
Client	MSG_SEND	SMR_MSG_SEND	..MSG_SEND00

- The client can send one or multiple messages in one transaction
- Each messages can have a different content and can be sent to different recipients
- Messages can be of a different type (page, SMS or email)
- Messages of the same type can be sent to different providers

Main Message

- Starts after the "[Header Delimiter](#)" (STX - \$2) character.
- Contains one or more message records.
- Each record consists of a message text and message data.
- Message text and data are separated by the [ETX](#) (\$3) control character.
- Message data contain different fields of information that the server needs to send the message.
- Fields are delimited with the "[Field Delimiter](#)" character.
- Multiple message records are delimited with the [RS](#) (\$1E) control character.
- The last record does not end with the RS character.

Example:

```
<STX>message text<ETX>message data<RS>
```

```
<STX>message text<ETX>message data<RS>
```

```
<STX>message text<ETX>message data
```

- If the Main Message contains only one record, it does not end with the RS character.

Example:

```
<STX>message text<ETX>message data
```

Message Text

- Size: variable size
- Position:

- Starts with the [Header Delimiter](#) character ([STX](#) - \$2)
- Ends with the message text delimiter character ([ETX](#) - \$3)
- Message Format: [Unicode String](#)
- Usage:
 - Contains a user's page, SMS or email message text.
 - It can be an empty string.
 - It can contain new line characters (CR - \$D, LF - \$A).
 - Only characters that are not allowed are [ETX](#) (\$3) and [RS](#) (\$1E).
 - It is never enclosed in any type of quotes. If it contains quotes, the server will consider them as a part of the message text and send them to the receiver.

Message Data

Fields

Position	Contains	Data Type	Required	Value Required
1	Message Id	Number	✓	x
2	Carrier Id	Number	✓	✓
3	Message Address	ASCII String	✓	✓
4	Message Priority	Number	x	x
5	Message Age	Number	x	x
6	Email Subject	Unicode String	x	x
Version 1.2				
7	Contact Name	Unicode String	x	x

- All **required** fields must be present.
- All fields with a **required value** must have a valid value and the value must be in a required format.
- If the non-required field is present, all fields before it (even if empty) must be present

1 Message Id

- A unique number that the client assigns to the message
- The server returns the same number with the send message result
- Can be empty or '0'

2 Carrier Id

- The system id of the carrier registered on the server
- If the server cannot find the carrier, it will return a send message error and the message will not be sent

3 Message Address

- A pager id, mobile number or email address of the receiver

- If the address is missing or is not properly formatted, the server will return a send message error and the message will not be sent
- The format of the address depends on the “Carrier Id” field – a value must be properly formatted to match the type of address requested by the carrier.

4 Message Priority

- The [priority of the message](#) in the server's message queue.
- Can be empty or missing – in that case the server will assign the default priority. Default: `MPR_NORMAL (4)`

5 Message Age

- A number of minutes the server will keep an undelivered message before it is discarded.
- Can be empty or missing – in that case the server will assign the default age. Default: `120` minutes (`2` hours)

6 Email Subject

- The subject in the email message header.
- Format: [Unicode String](#)
- Control characters `ETX ($3)` and `RS ($1E)` are not allowed.
- If the “[Field Delimiter](#)” character or words enclosed in double quotes are used, then the whole string must be enclosed in double quotes. See [Delimiting Strings](#)
- It can be empty or missing – in that case the server will assign the default email header, but only for email messages. Default: `'i-Page Message'`

7 Contact Name (V 1.2)

- the name of the contact which address is used in the “**Message Address**” field in the #3

Example – Message Data Fields:

Single Message

```
3611,3,1234567,4,194,i-Page Multi Message,IT Manager
```

Multi message

```
<msg>3609,3,1234567,4,194,i-Page Multi Message,IT Manager1E
<msg>3610,1,2643029,4,194,i-Page Multi Message,Test1E
<msg>3611,3,1234568,4,194,i-Page Multi Message,Nicole Pager1E
<msg>3612,4,sam@info.com,4,194,i-Page Multi Message,Sam Email1E
<msg>3613,2,+650214569871,4,194,i-Page Multi Message,Sam Mobile
```

Example – Complete Send Message:

Single message

```
11..MSG_SEND000000000,7DD6071338291F76BC1E0000001Dnicole2
Hello world from Nicole3611,3,1234567,4,194,i-Page Multi Message,IT Manager
```

Multi message

```
11..MSG_SEND000000000,7DD6071338291F76BC1E0000001Dnicole2
Test message for TAP Direct carrier from Nick3609,3,1234567,4,194,i-Page Multi
Message,IT Manager1E
```

```

Test message for TAP Modem carrier from Peter3610,1,2643029,4,194,i-Page Multi
Message,Test1E
Hello from Rachel3611,3,1234568,4,194,i-Page Multi Message,Nicole Pager1E
Test message for SMTP carrier from Sam3612,4,sam@info.com,4,194,i-Page Multi
Message,Sam Email1E
Test message for SMS carrier from Tom3613,2,+650214569871,4,194,i-Page Multi
Message,Sam Mobile

```

Send Message Result

On every message sent by the client using “[Send Message](#)” command, the server returns a result. If the client sends multiple messages in one transaction, the result is returned for every message separately.

The result is defined by the “Command Subtype” field.

Sender	Command	Subtype	Example
Server	MSG_SEND	SMR_MSG_SENT_OK	..MSG_SEND01
Server	MSG_SEND	SMR_**	..MSG_SEND** (02 - 19)

Header

- The “[Command Subtype](#)” field defines the send message result
 - If the message was sent to the carrier without errors the subtype field contains the [SMR_MSG_SENT_OK \(\\$01\)](#) code.
 - In all other cases this field contains an error code. Error codes range from [\\$02](#) to [\\$19](#). see “[Send Message Subtypes](#)”
 - Error codes further specify a cause of the error.
 - Error code [SMR_IPP_PROTOCOL_ERROR \(\\$19\)](#) is returned if neither i-Page Server nor a provider can determine a real cause of the error.
- The “[Message Id](#)” field contains the same transaction id that was sent by the client as a part of the sent message
 - If multiple messages were sent in the same transaction, then the [message id](#) from the [message data](#) part of the message record is returned
- The “[Date Received](#)” field contains a [DateTime](#) value when the server had received the message
- The “[Date Returned](#)” field contains a [DateTime](#) value when the server had sent the message to the carrier or returned an error to the client.

Main Message

- Starts after the “[Header Delimiter](#)” ([STX](#) - [\\$2](#)) character.
- Contains only one message record with a message text and message data
- Message text and data are separated by the [ETX](#) ([\\$3](#)) control character.
- Message data contain different fields of information about the sent message.

- Fields are delimited with the “[Field Delimiter](#)” character.

Example:

<STX>message text<ETX>message data

Message Text

- Size: variable size
- Position:
 - Starts with the [Header Delimiter](#) character (STX - \$2)
 - Ends with the message text delimiter character (ETX - \$3)
- Message Format: [Unicode String](#)

Usage:

- Contains a user friendly description of the result code from the “[Command Subtype](#)” field
 - ◇ If the subtype field has SMR_MSG_SENT_OK value, contains a confirmation that the message was sent successfully.
 - ◇ For all other SMR_* subtypes, contains a description of the error.
- It is never an empty string.
- It can contain new line characters (CR - \$D, LF - \$A).
- Only character that is not allowed is ETX (\$3).
- It is never enclosed in any type of quotes, but it can contain quoted words.

Message Data

Fields

Position	Contains	Data Type
1	Message Address	ASCII String
2	Address Type	Number
3	Original Message Text	Unicode String
Version 1.2		
4	Contact Name	Unicode String

- All fields are always returned by the server
- Some fields may not have value, which means that their value was not sent in the original message from the client

1. Message Address

- A pager id, mobile number or email address of the receiver that the message was sent to

2. Address Type

- Number constant that denotes a [type of the address](#)

3. Original Message Text

- The user's page, SMS or email message that was sent to the server
- Format: [Unicode String](#)
- Control character [ETX \(\\$3\)](#) is not allowed.
- If the "[Field Delimiter](#)" character or words enclosed in double quotes are used, then the whole string must be enclosed in double quotes. See [Delimiting Strings](#)
- It can be empty or missing, which means that the original message was an empty string.

4. Contact Name (V 1.2)

- The name of the contact that the message was sent to and which address is referenced in #1

Example:

```
11..MSG_SEND0100000265,7DD60C0B010C1F7DD60C0B01291Dnicole2Message sent OK.  
32643029,1,Test message for the TAP carrier,Test Pager
```

```
11..MSG_SEND1100000267,7DD60C0B043A1F7DD60C0B05061Dnicole2No login request from  
the carrier.31234568,1,Test message for the TAP Direct carrier from  
Rachel,Nicole Pager
```

```
11..MSG_SEND050000026D,7DD60C0C1A0F1F7DD60C0C1A0F1Dnicole2Device "GSM Modem" is  
disabled. ODOA Contact your administrator30273214569,2,"Hello, world from  
""WiPath"" and 'kiwi' programmers",IT Manager
```

Loading Objects

All system and message objects are stored on the server. The client can load and use these objects at any time during the client/server communication.

Following objects can be loaded:

1. Account – only the account that is currently logged on
2. System
3. Carrier – basics
4. Carrier – with object summary
5. Contact
6. Contact Group
7. Template
8. Schedule
9. Report
10. Message Settings – only message settings for the account that is currently logged on
11. Folder – Version 1.2

Loading objects is an optional operation and the client can successfully communicate with the server without doing it.

Load Request

To load any of the above objects, the client must send the “Load” (**LOAD**) command. The server recognises the type of the object the client is interested in by the “[Command Subtype](#)” field. Every object has a different subtype and it can be only one of the “**LOR_***” subtype codes.

For every object type the client wishes to load, a separate command must be issued.

Sender	Command	Subtype	Example	Object Type
Client	LOAD	LOR_CONTACTLOAD00	Contact
		LOR_GROUPLOAD03	Group
		LOR_TEMPLATELOAD06	Template
		LOR_ACCOUNTLOAD09	Account Settings
		LOR_SYSTEMLOAD0C	System Settings
		LOR_CARRIERLOAD0F	Carrier Basics

Sender	Command	Subtype	Example	Object Type
		LOR_CARRIER_SUMLOAD12	Carrier with Summary
		LOR_SCHEDULELOAD15	Schedule
		LOR_REPORTLOAD18	Report
		LOR_MSG_SETUPLOAD1B	Message Setup
		LOR_FOLDERLOAD24	Folder (V 1.2)

Example:

```
11.....LOAD0000002385,7DD6190E2A1C1F76BC1E0000001Dnicole2Loading contact
objects from i-Page Server
```

```
11.....LOAD0300002386,7DD6190E2A1C1F76BC1E0000001Dnicole2Loading group objects
from i-Page Server
```

Load Success

Most of time the server will respond on the “Load” command with one of the “**LOR*_OK**” subtype codes.

In that case the server sends the client all objects of that type that have been created by or assigned to the account currently logged on.

Sender	Command	Subtype	Example	Object Type
Client	LOAD	LOR_CONTACT_OKLOAD01	Contact
		LOR_GROUP_OKLOAD04	Group
		LOR_TEMPLATE_OKLOAD07	Template
		LOR_ACCOUNT_OKLOAD0A	Account Settings
		LOR_SYSTEM_OKLOAD0D	System Settings
		LOR_CARRIER_OKLOAD10	Carrier Basics
		LOR_CARRIER_SUM_OKLOAD13	Carrier with Summary
		LOR_SCHEDULE_OKLOAD16	Schedule
		LOR_REPORT_OKLOAD19	Report
		LOR_MSG_SETUP_OKLOAD1C	Message Setup
		LOR_FOLDER_OKLOAD25	Folder (V 1.2)

Main Message

- Contains records of data where each record corresponds to one object of the load type.
- Records are delimited by **RS (\$1E)** -- record separator ASCII control character.
- Each record is divided into fields that represent properties of the object.
- Fields in the record are delimited by the “**Field Delimiter**”, defined in the header.

- If the subtype of the “Load” message is **LOR_ACCOUNT_OK** the main message contains only one record with fields that represent properties of the account currently logged on. Account fields are also divided by the “[Field Delimiter](#)”, defined in the header. [RS](#) (record separator) is not used.
- Structure and position of the fields inside records are different for different object types and they depend on the “[Command Subtype](#)” field of the message. For further explanations see “[Object Fields](#)” section.

Example for:

```

.....LOADLOR_CONTACT_OK

11.....LOAD0100002385,7DD6190E2A1C1F7DD6190E2B391Dnicole
21,Sharon,nicole,0,2643029,1,1E
2,Nick,nicole,0,1234567,3,1E
3,Rachel,nicole,0,1234568,3,1E
4,IT Help,nicole,0,itdesk@company.com,4,1E
6,Thomas,nicole,0,+650214569871,2,1E
17,Office Info,nicole,0,+64211013654,2,Call this number for contact info1E
18,Ops Manager,admin,1,1234567,3,

.....LOADLOR_GROUP_OK

11.....LOAD0400002386,7DD6190E2A1C1F7DD6190E2C301Dnicole
21,Front Desk,nicole,0,"3,6,7,10",1E
2,Technicians,nicole,0,"2,8,9,14,19",On duty only after 16:001E
6>Contact Info,admin,1,"1,13,17,18",1E
7,Project Info,admin,1,"3,7,8,18",1E
8,Assignment Info,admin,1,"7,15,18",1E
12,IT Group,nicole,0,"1,8,12,14",IT Department contacts 0DOA Updated monthly
    
```

Load Error

If the server encounters an error while trying to send requested data, it will respond with one of the “[LOR *_ERROR](#)” command subtypes.

Sender	Command	Subtype	Example	Object Type
Client	LOAD	LOR_CONTACT_ERRORLOAD02	Contact
		LOR_GROUP_ERRORLOAD05	Group
		LOR_TEMPLATE_ERRORLOAD08	Template
		LOR_ACCOUNT_ERRORLOAD0B	Account Settings
		LOR_SYSTEM_ERRORLOAD0E	System Settings
		LOR_CARRIER_ERRORLOAD11	Carrier Basics
		LOR_CARRIER_SUM_ERRORLOAD14	Carrier with Summary
		LOR_SCHEDULE_ERRORLOAD17	Schedule
		LOR_REPORT_ERRORLOAD1A	Report
		LOR_MSG_SETUP_ERRORLOAD1D	Message Setup

Sender	Command	Subtype	Example	Object Type
		LOR_FOLDER_ERRORLOAD26	Folder (V 1.2)

Example:

11.....LOAD1400002387,7DD6190E2A1C1F7DD6190E2C301Dnicole
 2Error loading carrier objects from i-Page Server

11.....LOAD0B00002389,7DD6190E2A1C1F7DD6190E2B391Dnicole
 2Error loading user account data from i-Page Server

Manipulating Message Objects

The client can create a new [message objects](#) that are accessible only to the account that is currently logged on. The same account can also edit and delete these objects.

Note: A “Folder” type of object is introduced with the version 1.2.

New Object

All message objects are created on the server. To create a message object the client sends the “New Object” (**OBJ_NEW**) command to the server. The “[Command Subtype](#)” field defines the type of the message object that will be created. It can be one of the “[LOR_*](#)” subtype codes.

Command subtypes that can be used with the **OBJ_NEW** command and object types that can be created by the client:

Sender	Command	Subtype	Example	Object Type
Client	OBJ_NEW	LOR_CONTACT	...OBJ_NEW00	Contact
		LOR_GROUP	...OBJ_NEW03	Group
		LOR_TEMPLATE	...OBJ_NEW06	Template
		LOR_SCHEDULE	...OBJ_NEW15	Schedule
		LOR_REPORT	...OBJ_NEW18	Report
		LOR_FOLDER	...OBJ_NEW24	Folder (V 1.2)

Example:

```
11...OBJ_NEW0000002190,7DD6161111331F76BC1E0000001Dnicole2Creating a new contact object on i-Page Server
```

```
11...OBJ_NEW0600002191,7DD6161116021F76BC1E0000001Dnicole2Creating a new template object on i-Page Server
```

```
11...OBJ_NEW1500002192,7DD6161117041F76BC1E0000001Dnicole2Creating a new schedule object on i-Page Server
```

New Object Success

If the requested object is successfully created on the server, the client gets the message with object's data. The “[Command Subtype](#)” field defines the type of the message object and it is always one of the “[LOR*_OK](#)” codes.

Command subtypes and object types that can be returned with the **OBJ_NEW** command:

Sender	Command	Subtype	Example	Object Type
Server	OBJ_NEW	LOR_CONTACT_OK	...OBJ_NEW01	Contact
		LOR_GROUP_OK	...OBJ_NEW04	Group
		LOR_TEMPLATE_OK	...OBJ_NEW07	Template
		LOR_SCHEDULE_OK	...OBJ_NEW16	Schedule

Sender	Command	Subtype	Example	Object Type
		LOR_REPORT_OK	...OBJ_NEW19	Report
		LOR_FOLDER_OK	...OBJ_NEW25	Folder (V 1.2)

Main Message

- Contains fields with default property values for the newly created object.
- Fields are delimited by the “[Field Delimiter](#)”, defined in the header.
- Structure and position of the fields are different for different object types and they depend on the “[Command Subtype](#)” field of the message. For further explanations see “[Object Fields](#)” section.

Example:

```
11...OBJ_NEW0100002190,7DD6161111331F7DD6161114381Dnicole221,Contact
21,nicole,0,,1,.
11...OBJ_NEW0700002191,7DD6161116021F7DD6161116181Dnicole211,Template
11,nicole,0,Enter template text here,.
11...OBJ_NEW1600002192,7DD6161117041F7DD61611171E1Dnicole221,Schedule
21,nicole,0,2,3,0,10,1,4,120,0,7DD617111700,7DD702111700,,i-Page Server
message,i-Page Server scheduled message default,0,0,76BC1E000000,.
```

New Object Error

If the creation of the requested object fails on the server, the client gets an error message with one of “[LOR*_ERROR](#)” subtypes.

Command subtypes that can be returned as an error with the **OBJ_NEW** command:

Sender	Command	Subtype	Example	Object Type
Server	OBJ_NEW	LOR_CONTACT_ERROR	...OBJ_NEW02	Contact
		LOR_GROUP_ERROR	...OBJ_NEW05	Group
		LOR_TEMPLATE_ERROR	...OBJ_NEW08	Template
		LOR_SCHEDULE_ERROR	...OBJ_NEW17	Schedule
		LOR_REPORT_ERROR	...OBJ_NEW1A	Report
		LOR_FOLDER_ERROR	...OBJ_NEW26	Folder (V 1.2)

Example:

```
11...OBJ_NEW0200002190,7DD6161111331F76BC1E0000001Dnicole2Error creating a new
contact object on i-Page Server
11...OBJ_NEW0800002191,7DD6161116021F76BC1E0000001Dnicole2Error creating a new
template object on i-Page Server
11...OBJ_NEW1700002192,7DD6161117041F76BC1E0000001Dnicole2Error creating a new
```

schedule object on i-Page Server

Save Object

All data about message objects are kept on the server. To save changes to a message object, the client sends the “Save Object” (**OBJ_SAVE**) command to the server. The “[Command Subtype](#)” field defines the type of the message object that will be saved. It can be one of the “[LOR_*](#)” subtype codes.

The server does not return any response (neither failure nor success) for the “Save” command.

Command subtypes that can be used with the **OBJ_SAVE** command and object types that can be edited and saved by the client:

Sender	Command	Subtype	Example	Object Type
Client	OBJ_SAVE	LOR_CONTACT	..OBJ_SAVE00	Contact
		LOR_GROUP	..OBJ_SAVE03	Group
		LOR_TEMPLATE	..OBJ_SAVE06	Template
		LOR_SCHEDULE	..OBJ_SAVE15	Schedule
		LOR_REPORT	..OBJ_SAVE18	Report
		LOR_FOLDER	..OBJ_SAVE24	Folder (V 1.2)

Main Message

- ❑ Contains fields with edited property values for the message object.
- ❑ Fields are delimited by the “[Field Delimiter](#)”, defined in the header.
- ❑ Structure and position of the fields are different for different object types and they depend on the “[Command Subtype](#)” field of the message. For further explanations see “[Object Fields](#)” section.

Example:

```
11..OBJ_SAVE0000002336,7DD6180F342F1F76BC1E0000001Dnicole217,Office
Info,nicole,0,+64211013654,2,Call this number to get any contact info
```

```
11..OBJ_SAVE0300002338,7DD6181000281F76BC1E0000001Dnicole
22,Technicians,nicole,0,"2,8,9,14,19",On duty only after 16:00
```

```
11..OBJ_SAVE1500002347,7DD6190A391C1F76BC1E0000001Dnicole22,Sales
Update,nicole,0,2,3,0,73,3,4,120,112,7DD61A102B00,7DD809102B00,"1,2,3",i-Page
Server message,Reminder! Regular sales update due.,0,0,7DD118102B00,Regular
sales update 0DOA Send every 3 days
```

Delete Object

All message objects are kept on the server. To delete a message object the client sends the “Delete Object” (**OBJ_DEL**) command to the server. The “[Command Subtype](#)” field defines the type of the message object that will be deleted. It can be one of the “[LOR_*](#)” subtype codes.

The server does not return any response (neither failure nor success) for the “Delete” command.

Command subtypes that can be used with the **OBJ_DEL** command and object types that can be deleted by the client:

Sender	Command	Subtype	Example	Object Type
Client	OBJ_DEL	LOR_CONTACT	...OBJ_DEL00	Contact
		LOR_GROUP	...OBJ_DEL03	Group
		LOR_TEMPLATE	...OBJ_DEL06	Template
		LOR_SCHEDULE	...OBJ_DEL15	Schedule
		LOR_REPORT	...OBJ_DEL18	Report
		LOR_FOLDER	...OBJ_DEL24	Folder (V 1.2)

Main Message

- Contains ID of the message object that will be deleted

Example:

```
11...OBJ_DEL000000234A,7DD6190B0B021F76BC1E0000001Dnicole221
```

```
11...OBJ_DEL1500002348,7DD6190B08261F76BC1E0000001Dnicole216
```

Find Object

The client can search for the system-wide message objects according to criteria set by the user. The search is performed only on the message objects that are visible to all the accounts.

To search for the message objects the client sends the “Query” (**QUERY_RUN**) command to the server. The “[Command Subtype](#)” field defines the type of the message objects that will be searched for. It can be one of the “**LOR_***” subtype codes.

Command subtypes that can be used with the **QUERY_RUN** command and object types that can be searched for by the client:

Sender	Command	Subtype	Example	Object Type
Client	QUERY_RUN	LOR_CONTACT	.QUERY_RUN00	Contact
		LOR_GROUP	.QUERY_RUN03	Group
		LOR_TEMPLATE	.QUERY_RUN06	Template
		LOR_SCHEDULE	.QUERY_RUN15	Schedule
		LOR_REPORT	.QUERY_RUN18	Report
		LOR_FOLDER	.QUERY_RUN24	Folder (V 1.2)

Main Message

- Contains two fields with search criteria set by the user.

- [Search Type](#)
- Search String
- Fields are delimited by the “[Field Delimiter](#)”, defined in the header.

Example:

```
11.QUERY_RUN000000234B,7DD6190B30261F76BC1E0000001Dnicole21,office
```

```
11.QUERY_RUN030000234D,7DD6190B312D1F76BC1E0000001Dnicole22,info
```

Find Object Success

If the server finds one or more objects of the requested type that match user's criteria, it returns one of the “[LOR*_OK](#)” subtype codes with the “Query” (**QUERY_RUN**) command. The “[Main Message](#)” field contains data about found objects.

Command subtypes and object types that can be returned with the **QUERY_RUN** command:

Sender	Command	Subtype	Example	Object Type
Server	QUERY_RUN	LOR_CONTACT_OK	.QUERY_RUN01	Contact
		LOR_GROUP_OK	.QUERY_RUN04	Group
		LOR_TEMPLATE_OK	.QUERY_RUN07	Template
		LOR_SCHEDULE_OK	.QUERY_RUN16	Schedule
		LOR_REPORT_OK	.QUERY_RUN19	Report
		LOR_FOLDER_OK	.QUERY_RUN25	Folder (V 1.2)

Main Message

- Contains records of data where each record corresponds to one message object of the command subtype.
- Records are delimited by [RS \(\\$1E\)](#) -- record separator ASCII control character.
- Each record is divided into fields that represent properties of the object.
- Fields in the record are delimited by the “[Field Delimiter](#)”, defined in the header.
- If only one object is found [RS \(\\$1E\)](#) is not used.
- The last object's record does not end with [RS \(\\$1E\)](#).
- Structure and position of the fields inside records are different for different object types and they depend on the “[Command Subtype](#)” field of the message. For further explanations see “[Object Fields](#)” section

Example:

```
11.QUERY_RUN040000234D,7DD6190B312D1F7DD6190B39351Dnicole
27,Project Info,admin,1,"3,7,8,18",1E
6,Contact Info,admin,1,"1,13,17,18",1E
8,Assignment Info,admin,1,"7,15,18",
```

```
11.QUERY_RUN010000234E,7DD6190C05281F7DD6190C052F1Dnicole
```

```
215,Sam,admin,1,sam@info.com,4,1E
18,Ops Manager,admin,1,1234567,3,1E
13,Greg IT A,admin,1,+64022322365,2,
```

Find Object Error

If the server cannot find any object of the requested type that match user's criteria, it returns one of the “LOR_*_ERROR” subtype codes with the “Query” (QUERY_RUN) command.

Sender	Command	Subtype	Example	Object Type
Server	QUERY_RUN	LOR_CONTACT_ERROR	.QUERY_RUN02	Contact
		LOR_GROUP_ERROR	.QUERY_RUN05	Group
		LOR_TEMPLATE_ERROR	.QUERY_RUN08	Template
		LOR_SCHEDULE_ERROR	.QUERY_RUN17	Schedule
		LOR_REPORT_ERROR	.QUERY_RUN1A	Report
		LOR_FOLDER_ERROR	.QUERY_RUN26	Folder (V 1.2)

Example:

```
11.QUERY_RUN0200000E86,7DC11F0B382D1F7DC11F0B38321Dnicole
2Searching for "Contact" objects failed 0D0A There are no contacts that meet
search conditions. 0D0A Search conditions: Objects containing text "teacher"
```

```
11.QUERY_RUN0800000E87,7DC11F0B3A011F7DC11F0B3A051Dnicole
2Searching for "Template" objects failed 0D0A There are no templates that meet
search conditions. 0D0A Search conditions: Objects containing text "accident"
```

Assign Objects

The client can assign any of the system-wide message objects to the account currently logged on. Only the system message objects can be assigned to the account.

To assign the message objects, the client sends the “Change Account” (ACC_CHNG) command to the server. The command subtype defines the type of the message objects that will be assigned. It can be one of the “LOR_*” subtype codes. In this version only contacts, groups, templates, schedules and reports can be assigned.

The server does not return any response (neither failure nor success) for the “Change Account” command.

Sender	Command	Subtype	Example	Object Type
Client	ACC_CHNG	LOR_CONTACT	..ACC_CHNG00	Contact
		LOR_GROUP	..ACC_CHNG03	Group
		LOR_TEMPLATE	..ACC_CHNG06	Template
		LOR_SCHEDULE	..ACC_CHNG15	Schedule
		LOR_REPORT	..ACC_CHNG18	Report
		LOR_FOLDER	..ACC_CHNG24	Folder (V 1.2)

Main Message

- Contains IDs of all the message objects of that type that are assigned to the account.
- ID data type is [Number](#).
- All IDs are delimited with the “[Field Delimiter](#)”, defined in the header.
- When the client send assigned objects to the server, main message must always contain all assigned objects of the type.

Example:

```
11..ACC_CHNG000000E9A,7DC11F0D282C1F76BC1E0000001Dnicole  
278,77,60,80,12,30,79,38,9,59,86,52,53,81,87,89,90,54,57,73,61,65,67,66
```

```
11..ACC_CHNG0300000E9D,7DC11F0D370B1F76BC1E0000001Dnicole  
21,2,3,4,5
```

Other Messages

Change Password

The client can send a message to the server with the request to change the user's password. Only the password of the account currently logged on through the client can be changed that way.

The client sends the “[Login](#)” (LOGIN) command with the “[LR_CHANGE_LOGIN](#)” command subtype.

Sender	Command	Subtype	Example
Client	LOGIN	LR_CHANGE_LOGINLOGIN07

Main Message

- Contains 4 fields with the account's login information:
- Fields are delimited with the “[Field Delimiter](#)”, defined in the header.
- Fields:
 - Old Password
 - New Password
 - New Password Confirmed
 - [Encoding type](#)
- The “Old Password”, “New Password” and “New Password Confirmed” fields can be encoded or in plain text.
- First 3 fields must be present.
- The encoding type field can be empty or missing. In that case the server will consider all fields to be a plain text.
- To perform encoding, the client needs to use “*IPCrypt.dll*” library.

Example:

```
11.....LOGIN0700000E88,7DC11F0C161E1F76BC1E0000001Dnicole
2nicole,nicky,nicky,0
```

```
11.....LOGIN0700000B12,7DC1190B39101F76BC1E0000001Dnicole
2nicole,2FC48777E19031B20A2D469879DB9C90,2FC48777E19031B20A2D469879DB9C90,1
```

Change Password Success

If the password has been changed successfully the server returns the “[Login](#)” (LOGIN) command

with the “[LR_CHANGE_LOGIN_OK](#)” command subtype

Sender	Command	Subtype	Example
Server	LOGIN	LR_CHANGE_LOGIN_OKLOGIN08

Example:

```
11.....LOGIN0800000E88,7DC11F0C161E1F7DC11F0C170E1Dnicole
2Password for "Nicole" successfully changed
```

Change Password Error

If the password has not been changed successfully the server returns the “[Login](#)” (LOGIN) command with the “[LR_CHANGE_LOGIN_ERROR](#)” command subtype

Sender	Command	Subtype	Example
Server	LOGIN	LR_CHANGE_LOGIN_ERRORLOGIN09

Example:

```
11.....LOGIN0900000E96,7DC11F0C303A1F7DC11F0C31381Dnicole
2Changing password for "Nicole" failed. 0D0A Old password not successfully
confirmed
```

```
11.....LOGIN0900000E97,7DC11F0C32371F7DC11F0C32381Dnicole
2Changing password for "Nicole" failed. 0D0A New password not successfully
confirmed
```

Change Account Settings

To change the settings of the account currently logged on, the client sends the “[Change Account](#)” (ACC_CHNG) command to the server with the “[LOR_ACCOUNT](#)” subtype code.

The server does not return any response (neither failure nor success) for the “Change Account” command.

Sender	Command	Subtype	Example
Client	ACC_CHNG	LOR_ACCOUNT	..ACC_CHNG09

Main Message

- Contains data about the account that has been changed by the client and sent to the server.
- The account properties are send as fields that are delimited with the “[Field Delimiter](#)”, defined in the header.
- For the field structure see “[Account – Fields](#)”.
- The first field, an account ID, must belong to the account that is logged on the client and registered on the server

Example:

```
11..ACC_CHNG0900000E98,7DC11F0D102B1F76BC1E0000001Dnicole
2nicole,131071,1234567,+640211013226,nicky@server.com,Nicole
```

Run Report

The client can run any report that is created or assigned to the account currently logged on. The report is run on the server and the result is returned as a CSV file attachment to emails defined in the report settings.

To run the report the client sends the “Query” ([QUERY_RUN](#)) command to the server with [LOR_RUN_REPORT](#) command subtype..

Sender	Command	Subtype	Example
Client	QUERY_RUN	LOR_RUN_REPORT	.QUERY_RUN1E

Main Message

Contains the system ID of the report to be run.

Example:

```
11.QUERY_RUN1E00001DEE,7DD4120D12111F76BC1E0000001Dnicole22
```

Run Report Success

If the report has been run successfully the server returns the “Query” ([QUERY_RUN](#)) command with the [LOR_RUN_REPORT_OK](#) command subtype

Sender	Command	Subtype	Example
Server	QUERY_RUN	LOR_RUN_REPORT_OK	.QUERY_RUN1F

Example:

```
11.QUERY_RUN1F00001DF0,7DD4120D18211F7DD4120D182A1Dnicole
2Report "Office AUCK Report" with id "2" is running on i-Page Server. 0D0A Its
result will be send as an email attachment to: 0D0A predrag@wipath.com
```

Run Report Error

If the report has not been run successfully the server returns the “Query” ([QUERY_RUN](#)) command with the [LOR_RUN_REPORT_ERROR](#) command subtype

Sender	Command	Subtype	Example
Server	QUERY_RUN	LOR_RUN_REPORT_ERROR	.QUERY_RUN20

Example:

```
11.QUERY_RUN2000001DFD,7DD4120D1E371F7DD4120D1E3B1Dnicole
2Report with id "25" not found on i-Page Server.
```

Query Sent Messages

All sent messages from all the accounts are stored on the server. The client can query the server to return records of all sent messages that match certain criteria defined by the user.

To run the query the client sends the “Query” ([QUERY_RUN](#)) command to the server with [LOR_SENT_MSG](#) command subtype..

Sender	Command	Subtype	Example
Client	QUERY_RUN	LOR_SENT_MSG	.QUERY_RUN21

Main Message

- Contains 4 fields with query criteria
- Fields are delimited with the “[Field Delimiter](#)”, defined in the header.
- Fields
 - [Sent Messages Query Type](#) constant
 - A number of messages
 - ◇ used with [SMT_LAST](#)
 - “From date” field – a date or start of an interval
 - ◇ used with [SMT_TODAY](#), [SMT_DATE](#), [SMT_BEFORE](#), [SMT_AFTER](#), [SMT_BETWEEN](#)
 - “To date” field – end of interval
 - ◇ used with [SMT_TODAY](#), [SMT_BETWEEN](#)
- For the option [SMT_TODAY](#) (messages sent today), “From date” and “To date” options are both today’s date
- For options [SMT_ALL](#), [SMT_LAST](#), [SMT_DATE](#), [SMT_BEFORE](#) and [SMT_AFTER](#) “To date” field value is disregarded by the server

Example:

```
11.QUERY_RUN1E00001E0B7DD4120D330876BC1E000000,17nicole,0A3A02C71D23,2,7DD11700000,7DD116000000
```

```
11.QUERY_RUN1E00001E0C7DD4120D353876BC1E000000,17nicole,0A3A02C71D21,10,7DD117000000,7DD116000000
```

Query Sent Messages Success

If any message that match criteria has been found, the server returns the “Query” ([QUERY_RUN](#)) command with the [LOR_SENT_MSG_OK](#) command subtype

Sender	Command	Subtype	Example
Server	QUERY_RUN	LOR_SENT_MSG_OK	.QUERY_RUN22

Main Message

- ❑ Contains records of data where each record corresponds to one sent message that match selected criteria
- ❑ Records are delimited by [RS \(\\$1E\)](#) -- record separator ASCII control character.
- ❑ Each record is divided into fields that represent different information for the sent message.
- ❑ Fields in the record are delimited by the “[Field Delimiter](#)”, defined in the header.
- ❑ If only one sent message was found [RS \(\\$1E\)](#) is not used.
- ❑ The last sent message record does not end with [RS \(\\$1E\)](#).
- ❑ Structure and position of the fields inside are defined in the “[Object Fields](#)” section

Example:

```
11.QUERY_RUN2200001E0D,7DD4120D370D1F7DD4120D37111Dnicole
2315,82,2,2,1,3,12,3,0,1,4,2643029,"Please, come to my office. OD0A We are all
here.",Message is too long and had to be fragmented. OD0A No of fragments was
too big (3) and message was truncated to 2 fragments.
,nicole,127.0.0.1,TAP Modem,7DD118090E15,7DD118090E30,1,0,0,11E
314,81,1,1,1,1,1,3,0,1,4,2643029,"Please, come to my office. OD0A We are all
here.",Message sent OK.,
nicole,127.0.0.1,TAP Modem,7DD118090B39,7DD118090C18,1,0,0,01E
313,141,1,1,1,1,1,0,0,1,4,2643029,Nicole OD0A Hello girl. Waiting for you OD0A
24/01/2013 09-09-52,Message sent OK.,
nicole,10.58.2.199,TAP Modem,7DD118090935,7DD118090A13,1,0,0,01E
312,79,1,-1,1,1,1,3,0,1,4,2643029,"Please, come to my office. OD0A We are all
here.",Message sent OK.,nicole,127.0.0.1,,7DD118090231,7DD118090536,1,0,0,01E
311,78,1,0,3,2,5,3,1,1,4,1236547,Come to base ASAP,"Device ""Site Page"" is
disabled. OD0A Contact your administrator.",
nicole,127.0.0.1,TAP Direct,7DD118082E1E,7DD118082E1E,1,0,0,01E
307,83,1,1,2,2,14,3,0,2,4,0273214569,Important! Meeting today at 2:15 p.m. in my
office,"Device ""COM5"" could not be initialised.",
nicole,127.0.0.1,,7DD1170A2F26,7DD1170D1A3B,1,0,0,0
```

Query Sent Messages Error

If no message that match criteria has been found, the server returns the “Query” ([QUERY_RUN](#)) command with the [LOR_SENT_MSG_ERROR](#) command subtype

Sender	Command	Subtype	Example
Server	QUERY_RUN	LOR_SENT_MSG_OK_ERROR	.QUERY_RUN23

Example:

```
11.QUERY_RUN2300001E0F,7DD4120D382B1F7DD4120D382F1Dnicole2There is no message
for the required criteria
```

Object Fields

Object fields are used to exchange data about i-Page objects between the client and the server. Depending on the command and the command subtype, they can be sent either by the server or by the client. To make this exchange successful, several rules must be followed:

- ❑ All object fields are always sent in the “[Main Message](#)” part of the message.
- ❑ Structure and position of the fields for a particular object type is the same in every message and does not depend on the type of the message (message command and subtype).
- ❑ All fields are delimited by the “[Field Delimiter](#)” character, defined in the “Header”.
- ❑ If the message contains more than one object record, than the records are delimited by the [RS\(\\$1E\)](#) character.

System Objects

Account

The account object is used by the client to load and change the settings of the account that is currently logged on.

Warning: Do not try to change settings for other accounts, because it can have unpredictable results.

Commands

Sender	Command	Command Subtype
Server	LOAD	LOR_ACCOUNT_OK
Client	ACC_CHNG	LOR_ACCOUNT

Fields

Position	Contains	Data Type	Required
1	Account Id	String	✓
2	Permissions	Number	x
3	Pager Id	String	x
4	Mobile Number	String	x
5	Email Address	String	x
6	Display Name	String	x

- ❑ The server always sends all fields
 - The **Account Id** and **Permissions** fields are guaranteed to have valid values

- All other fields will have values if they are set on the server. Otherwise, they will be empty.
- If sent by the client, the only field that must be present and have a valid value is the “Account id” field
 - All other fields can be empty or omitted.
 - If the value for the field is set, all other fields that precede it must be included, even if they are empty.
 - Any field with an empty value will delete the existing value of its respective property on the server.
 - ◇ This does not affect the **Account Id** and **Permissions** fields, because they cannot be changed by the client.

Field Details

- 1 Account Id** – the id of the account that is logged on, as registered on the server.
 - 1.1** It must be present and have a valid value
- 2 Permissions** – a number in which every bit is mapped to one of permissions, defined in the system.
 - 2.1** If a bit is turned on, that means that the account has a respective permission.
 - 2.2** If all permissions are set, the number is \$FFFFFF or #1,048,575
 - 2.3** If sent by the server, the field is guaranteed to have a valid value.
 - 2.4** If sent by the client, the field does not need to have a value, because the server does not check it. That is because the client is not allowed to set permissions.

Bit	Permission	Description
0	Log On	Allows account to log directly on the server (not through a client)
1	Set Server	Allows account to change some server settings (type names and logs)
2	Set Accounts	Allows account to create, edit and delete accounts.
3	Set Devices	Allows account to create, set and delete system devices.
4	Set Carriers	Allows account to create, set and delete system carriers.
5	Set Database	Allows account to change the system database file.
6	Set System Message Objects	Allows account to create, edit and delete system wide contacts, groups, folders and templates on i-Page Server.
7	Set Client Message Objects	Allows accounts to create, edit and delete their own contacts, groups, folders and templates on i-Page Client
8	Assign System Message Objects	Allows account to assign system-wide contacts, groups, folders and templates and templates to themselves.

Bit	Permission	Description
9	Set Server Connection	Allows account to change server TCP/IP connection settings on the server.
10	Delete Server Message	Allows account to delete all sent messages stored on the server.
11	Delete Client Messages	Allows account to delete a subset of sent messages, stored on the server, that belongs to that account.
12	Register Server	Allows account to register the server and change the registration type (number of connections).
13	Manage Server Connections	Allows account to see details of all clients connected to the server and to disconnect any of them.
14	Manage Password Changes Log	Allows account to see all password changes to all accounts and save them to a file.
15	Set File Interface	Allows account to set, start and stop client file interface on the server
16	Set System Schedules	Allows account to create, edit and delete system wide schedules on i-Page Server
17	Set Account Schedules	Allows accounts to create, edit and delete their own schedules on i-Page Client
18	Set System Reports	Allows account to create, edit and delete reports on all accounts' sent messages on i-Page Server
19	Set Account Reports	Allows accounts to create, edit and delete reports but only on their own sent messages.

3 Pager Id – Account's pager id

4 Mobile number – Account's mobile number

5 Email address – Account's email address

6 Display name – A name that the system uses for that account for display and logging purposes

6.1 If not set, it defaults to the account id.

Example:

All fields are present

```
2nicole,131071,12345678,+640211013224,nicky@server.com,Nicole
```

Permissions and pager id fields are present but empty

```
2nicole,,,+640211013224,nicky@server.com,Nicole
```

Whole message example

```
11.....LOAD0A00002381,7DD6190E29101F7DD6190E29271Dnicole2nicole,786431,1234567,0213219875,nicky@server.com,Nicole
```

System

The System object is used to set common system type names for message objects on all clients. Common names are defined on the server by the system administrator.

Commands

Sender	Command	Command Subtype
Server	LOAD	LOR_SYSTEM_OK

Fields

Record No	Field No	Contains	Data Type
1	1	Contact Singular	String
	2	Contact Plural	String
2	1	Group Singular	String
	2	Group Plural	String
3	1	Template Singular	String
	2	Template Plural	String
4	1	Schedule Singular	String
	2	Schedule Plural	String
5	1	Report Singular	String
	2	Report Plural	String
Version 1.2			
6	1	Folder Singular	String
	2	Folder Plural	String

- The System object data are grouped into 6 records
- All Records, except the last one, are delimited by the [RS \(\\$1E\)](#) character.
- Each record contains 2 fields, delimited by the “[Field Delimiter](#)” character, defined in the “[Header](#)”.
- All fields in all the records are guaranteed to be present and have valid values.

1. **Singular** – singular case for the type name

2. **Plural** – plural version for the type name

Example:

2Contact, Contacts1E

Group, Groups1E

Template, Templates1E

Schedule, Schedules1E

Report, Reports1E

Folder, Folders

11.....LOAD0D00002382,7DD6190E29101F7DD6190E2A1E1Dnicole

2Contact, Contacts1E

Group, Groups1E

Template, Templates1E

Schedule, Schedules1E

Report, Reports1E

Folder, Folders

Carrier

The carrier object is used by the client to load information about all carriers available on the server.

The client requests from the server data about all carriers currently available on the server by sending the command combination **LOAD LOR_CARRIER (LOAD0F)**

If the server responds with the command combination **LOAD LOR_CARRIER_OK (LOAD10)** then the "[Main Message](#)" field contains data about all carriers registered on the server.

Command

Sender	Command	Command Subtype
Server	LOAD	LOR_CARRIER_OK
Server	LOAD	LOR_CARRIER_SUM_OK

Fields

Position	Contains	Data Type
1	Carrier Id	Number
2	Carrier Name	String
3	Carrier Type	Number
4	Address Type	Number
5	Message Length	Number
6	Carrier Object Summary	String

- All the fields are present and have valid values.
- The "Carrier Object Summary" field is present only if the command subtype is LOR_CARRIER_SUM_OK.
- To get the object summary, the client in its request must send LOR_CARRIER_SUM as the command subtype.

Field Details

- 1 Carrier Id** – the i-Page System id of the carrier as registered on the server.
- 2 Carrier Name** – the i-Page System name of the carrier as registered on the server
- 3 Carrier Type** – [carrier type](#) system constant
- 4 Address Type** – the [type of the address](#) that the carrier uses
- 5 Message Length** – the message length in bytes, as defined by the carrier
- 6 Carrier Object Summary** – a string with the summary of all the carrier properties
 - 6.1** The string is enclosed in double quotes and all double quotes used inside the string

are duplicated – see [“Delimiting Strings”](#)

Example:

```
21,TAP Modem,0,TAP Modem,1,pager id,160,"Carrier Id: 1,Carrier
Name: TAP Modem,Enabled: Yes,Carrier Type: TAP Modem,Address
Type: Pager id,Device Id: 1,Device Name: TAP Modem,Message
Length: 160 characters,Min Message Length: 18
characters,Preserves Line Breaks: Yes,Fragmenting: ODOA-
Fragmenting Allowed : Yes ODOA - Fragment Header : (Part 1 of 2)
ODOA- Maximum Number of Fragments: 9 ODOA - No Fragmenting
Action: Return error ODOA - To Many Fragments Action: Send
truncated,Comm Port Settings: ODOA- Baud Rate: 1200 ODOA - Byte
Size: 7 ODOA - Parity: Even ODOA - Stop Bits: 1,Uses
Authentication: No,Carrier Phone No: 0264001283"1E
```

Message Settings

The “Message Settings” object defines client system settings used for sending, querying, deleting and displaying messages and “[Message Objects](#)”. These settings are associated with the client's account currently logged on, because every account can personalise the client's program settings.

Commands

They are used with following commands:

Sender	Command	Command Subtype
Server	LOAD	LOR_MSG_SETUP_OK
Client	OBJ_SAVE	LOR_MSG_SETUP

Fields

- All fields must be present, but
- Only the “Owner Id” field must have a valid value.
 - The value of that field must be an account name registered on the server.
- All other fields can have an empty value.
 - In that case their system default will be used instead.

Position	Contains	Data Type
1	Owner Id	String
2	Message Priority	Number
3	Message Age	Number
4	Email Subject	String
5	Text Affix	String
6	Date Affix	String
7	Text Affix Type	Number
8	Date Affix Type	Number
9	Sent Messages Query Type	Number
10	Last <number> of Sent Messages	Number
11	Last Message Id	Number
12	Sent Messages <date> or <from_date>	DateTime
13	Sent Messages <to_date>	DateTime
14	Contact Display Type	Number
15	Sent Message Delete Type	Number

Position	Contains	Data Type
16	Delete Old Messages Interval	Number
17	Display Options	Number
Version 1.2		
18	Tree View Display Type	Number
19	Prefix Names Delimiter	String

Field Details

- 1 Owner Id** – ID of the account that the settings belong to
 - 1.1 It must contain a valid user name of the account registered on the server
- 2 Message Priority** - The priority of the message in the server's message queue
 - 2.1 The value must be in a range defined in "[Message Priorities](#)"
 - 2.2 It can be empty
 - 2.3 If it is out of the range or empty, the system will use a default
Default: MPR_NORMAL (4)
- 3 Message Age** – the time the server will keep the message in the message queue
 - 3.1 The time is expressed in minutes
 - 3.2 The server will keep the message if it did not manage to send it to a carrier because:
 - the server was down for a while
 - the message is always pushed back by messages with higher priority
 - sending to the carrier was not successful but the server did not use up all the trials set by the user
 - 3.3 It can be empty. In that case the system will use the default value
Default: 120 minutes
- 4 Email Subject** – the subject of the email message
 - 4.1 Used only in the email type of messages
 - 4.2 It can be empty. In that case the system will use the default value
Default: 'i-Page Message'
- 5 Text Affix** – constant text that will be added to every message
 - 5.1 The text can be added as a message prefix, suffix or both
 - 5.2 It can be empty. In that case nothing will be added to the message, regardless of the selected option in the "[Message Affix Types](#)"
 - 5.3 If you put one or more spaces before or after the text it will be preserved in relation to the message
 - 5.4 To enter a new line before or after the text, use "\n"
- 6 Date Affix** – sets formatting for adding a date to the message
 - 6.1 To format a date affix, use <DT> string as a place-holder for a date and time.
 - 6.2 If you put one or more spaces before or after the place-holder, it will be preserved in relation to the message
 - 6.3 To enter a new line before or after the place-holder, use "\n"
 - 6.4 It can be empty. In that case the date will be added to the message, immediately

before or/and after the message text, depending on the selected option in the [“Message Affix Types”](#)

- 7 Text Affix Type** – defines the position of the “Text Affix” in the message – for allowed values see [“Message Affix Types”](#)
 - 7.1** It can be empty. In that case the system will use the default value.
Default: MAT_NONE
- 8 Date Affix Type** – defines the position of the “Date Affix” in the message – for allowed values see [“Message Affix Types”](#)
 - 8.1** It can be empty. In that case the system will use the default value.
Default: MAT_NONE
- 9 Sent Messages Query Type** – type of the inquiry about sent messages
 - 9.1** For allowed types see [“Sent Messages Query Type”](#)
- 10 Show Number of Messages** – a number of sent messages used in the query
 - 10.1** Used if the option from #9 is SMT_LAST
 - 10.2** It can be empty. In that case the system will use the default value
Default: 10
- 11 Last Message Id** – the last id of the message that the account has sent
 - 11.1** This is a transaction id that the user can assign to every sent message. It allows the account to keep track of the sent messages, even if logged on the different machine
 - 11.2** It can be empty. In that case the counting will be reset to 0.
- 12 Sent Messages <date> or <from_date> -**
 - 12.1** A <date> is used for [Sent Messages Query Type](#) constants: SMT_DATE, SMT_BEFORE, SMT_AFTER
 - 12.2** <from_date> (start of interval) is used for [Sent Messages Query Type](#) constants: SMT_TODAY and SMT_BETWEEN
 - 12.3** For SMT_TODAY it contains today's date
- 13 Sent Messages <to_date> -** the end of the sent messages query period
 - 13.1** Used for [Sent Messages Query Type](#) constants: SMT_TODAY and SMT_BETWEEN
 - 13.2** For SMT_TODAY it contains today's date
- 14 Contact Display Type** – the user can select the way all contacts and groups are displayed. For allowed values see [“Contact Display Type”](#) - **Deprecated** – not used in [version 1.2](#)
- 15 Message Delete Type** – when deleting sent messages, the user can select between three options. For allowed values see [“Message Delete Type”](#)
- 16 Delete Old Messages Interval** – a number of days
 - 16.1** If MD_OLD constant is selected in the option #15, all sent messages older than <number_of_days> will be deleted
- 17 Display Options** – array of boolean display settings values
 - 17.1** All values are mapped to their respective bits in a number
 - 17.2** The number is then converted to 3 hexadecimals

17.3 Maximum value (if all bits are set) is:

17.3.1 Version 1.1 - \$7FF or #2,047

17.3.2 Version 1.2 – \$1FFF or #8,191

17.4 If all bits are not set, the hexadecimal value is left padded with '0'

Bit	Option	Description
0	On Send Clear Message	Clear the message text in display after the message was sent
1	On Send Clear Contacts	Clear all contacts in display after the message was sent
2	Allow Empty Message	Allow sending messages without text
3	Show System Contacts	Show system-wide contacts assigned to the account
4	Show System Groups	Show system-wide groups assigned to the account
5	Show System Templates	Show system-wide templates assigned to the account
6	Show System Schedules	Show system-wide schedules assigned to the account
7	Show System Reports	Show system-wide reports assigned to the account
8	Show Contacts First	If “ Contact Display Type ” is “Display Contacts and Groups” (CDT_BOTH), then display contacts first
9	Use Same Message	Use the same message for all selected contacts
10	Allow Duplicate Contacts	The same contact can be selected and used in the same transaction.
Version 1.2		
11	Add Object Names As Prefix	Every contact and group that the message is sent to is added to the send list as a prefix to the message
12	Show System Folders	Show system-wide folders assigned to the account

18 Tree View Display Type – (Replaces “Contact Display Type” in **V 1.2**) sets all objects that are displayed on the send message page and the order in which they are displayed

18.1 For every object 3 bits are set

18.2 First 3 bits (0..2) are reserved for contacts, second 3 bits (3..5) for groups and third 3 bits (6..8) for folders

18.3 First 2 bits (LSB) inside a respective object group are reserved for its position – 1st (01), 2nd (10), 3rd (11)

18.4 Third bit in the respective object group of bits represents its visibility. If the bit is set then the object is displayed.

Example:

(MSB)011 110 101(LSB) – contact, displayed on position 1, group displayed on position 2, folder is not displayed and it is on position 3

MSB)101 110 111(LSB) – all objects are displayed – folder on position 1, group on position 2 and contacts on position 3

19 Prefix Names Delimiter - (V 1.2) if the user selects to add every contact and group that the message is sent to as a prefix to the message text, this field sets the delimiter that will separate those entries.

Example:

```
11.....LOAD1C00002384,7DD6190E2A0A1F7DD6190E2B151Dnicole2nicole,4,194,i-Page
Multi Message,From Nicole: ,\n<DT>,0,0,2,10,399, 7DD615000000,7DD615000000,0,
1,45,5F8,382,:
```

Sent Message

The “Sent Message” object contains fields that the server stores for every sent message. It is returned as a result of the “[Query Sent messages](#)” command.

Command

It is used with the following command:

Sender	Command	Command Subtype
Server	QUERY_RUN	LOR_SENT_MSG_OK

Fields

All fields are returned by the server.

Position	Contains	Data Type
1	Message Id	Number
2	Transaction Id	Number
3	Number of Fragments	Number
4	Number of Sent Fragments	Number
5	Carrier Id	Number
6	Sent Status	Number
7	Sent Result Code	Number
8	Connection Type	Number
9	Carrier Type	Number
10	Address Type	Number
11	Message Priority	Number
12	Message Address	String
13	Message Text	String
14	Sent Result Description	String
15	Account Id	String

Position	Contains	Data Type
16	Client IP	String
17	Carrier Name	String
18	Date Received	DateTime
19	Date Sent	DateTime
20	Acknowledged	Boolean
21	Scheduled	Boolean
22	Escalated	Boolean
23	Fragmented	Boolean
Version 1.2		
24	Contact Name	String

Field Details

- 1 Message Id** – an id of the message assigned by the server
It is unique across the i-Page system.
- 2 Transaction Id** – a number that is assigned to the message by its sender at the time of sending
- 3 Number of Fragments** – the number of fragments the message is fragmented into
- 4 Number of Sent Fragments** – the number of fragments that were actually sent.
- 5 Carrier Id** – the i-Page System id of the carrier that the message was sent to
- 6 Sent Status** – the code of the sent status

Sent Status	Constant	Value
Not Sent	MSS_NOT_SENT	0
Sent	MSS_SENT	1
Sent Error	MSS_ERROR	2
Sent With Error	MSS_SENT_WITH_ERROR	3

- 7 Sent Result Code** – one of the “[Send Message Subtypes](#)” that more precisely defines result of the send message operation or error that might have occurred.
- 8 Connection Type** – the connection type to the server that the client was using to send the message.

Connection Type	Constant	Value
Local (COM)	CT_COM	0
TCP/IP	CT_TCP	1

File System	CT_FILE	2
Web	CT_WEB	3
Any	CT_ANY	4

8.1 Type **CT_ANY** is used mostly with scheduled messages, because the message is actually scheduled and sent directly from the server, after the user had defined the schedule settings.

8.2 It is also used in vary rare cases where the server was not able to define a type of connection of the message.

9 Carrier Type – one of the [carrier types](#) the message was sent to

10 Address Type – the [type of the address](#) used to send the message

11 Message Priority – a constant that defines a priority of the message in the server message queue – for the constant values see “[Message Priorities](#)”

12 Message Address – the address of the receiver

- Pager id
- Mobile number
- Email address

13 Message Text – the actual text of the user's message

13.1 If the text contains the “[Field Delimiter](#)” character or one or more words in double quotes, it will be enclosed in double quotes. For more see “[Delimiting Strings](#)”

14 Sent Result Description – the user friendly description of the send message result code from **#7**

14.1 If the text contains the “[Field Delimiter](#)” character or one or more words in double quotes, it will be enclosed in double quotes. For more see “[Delimiting Strings](#)”

15 Account Id – the server's id of the account that had sent the message

16 Client IP- IP address of the machine from which the message was sent

17 Carrier Name - the i-Page System name of the carrier as registered on the server at the time the message was sent

18 Date Received – date and time when the server has received the message – for the format see [DateTime](#) format

19 Date Sent – date and time when the server has sent the message to the carrier or returned an error to the client– for the format see [DateTime](#) format

20 Acknowledged – a [boolean](#) value that tells whether the server has returned message result to the client

21 Scheduled – a [boolean](#) value that tells whether the message was a scheduled message

22 Escalated – a [boolean](#) value that tells whether the message was an escalated message – in this version always '0'

23 Fragmented – a [boolean](#) value that tells whether the message was fragmented

24 Contact Name – (V 1.2) a name of the contact that the message was sent to

Example:

```
2315,82,2,2,1,3,12,3,0,1,4,2643029,"Please, come to my office. 0D0A We
are all here.",Message is too long and had to be fragmented. 0D0A No
of fragments was too big (3) and message was truncated to 2
fragments.,nicole,127.0.0.1, TAP Modem,7DD118090E15,7DD118090E30,
1,0,0,1,IT Manager1E
```

Message Objects

Common Properties

Message objects are the objects that the user uses to create and query messages. They include:

1. Contacts
2. Contact Groups
3. Message Templates
4. Schedules
5. Reports

Commands

They are used with following commands:

Sender	Command	Command Subtype
Server	LOAD	LOR_*_OK
Server	OBJ_NEW	LOR_*_OK
Client	OBJ_SAVE	LOR_*
Server	QUERY_RUN	LOR_*_OK

Fields

There are 5 fields that are common to all message objects:

Contains	Data Type	Required
Object Id	Number	✓
Object Name	String	x
Account Owner	String	✓

Contains	Data Type	Required
System Object	Boolean	✓
Notes	String	x

Field Details

- 1 **Object Id** – unique id assigned by the system when the object is created.
 - 1.1 It is unique in the set of the objects of the same type.
 - 1.2 It cannot be changed during the object's lifetime. After the object is deleted from the system, its id can be reused.
- 2 **Object Name** – descriptive name assigned by the creator of the object
 - 2.1 It can be changed by the account that created the object
 - 2.2 It is not unique and two or more objects of the same type can have the same name.
 - 2.3 It can contain spaces and any other printable characters compatible with the Unicode Basic Multilingual Plane.
- 3 **Account Owner** – the account that has created the object
 - 3.1 Objects that are created by the client application are visible only to the account that has created them. They can be changed or deleted only by the same account.
 - 3.2 Objects that are created by the server application are system-wide objects and can be visible to all accounts. By default their owner is the 'admin' account.
- 4 **System Object** – If true, indicates that the object is a system-wide object.
- 5 **Notes** – contains additional explanations, remarks or comments about the object.
 - 5.1 One or more lines of free flowing text
 - 5.2 It can contain new line characters ([\\$ODOA](#)).
 - 5.3 It can contain any other printable characters compatible with the Unicode Basic Multilingual Plane.
 - 5.4 It can be displayed in multi lane windows GUI controls (e.g. Memo, Rich Edit, etc.).

Application

There are several rules for field usage that are common to all message objects:

1. All fields of the object must be present.
2. Fields **Object Id**, **Account Owner** and **System Object** must have valid values. The client should obtain their values from the server and use it in subsequent messages to the server.

3. Sending incorrect values for these fields to the server, will not change the object's respective properties, but can render unpredictable results and behaviour.
4. All other fields can be empty.
5. If the client returns an empty value in any of the non-compulsory fields, the object's respective property on the server will be deleted.
 - This does not affect **Object Id**, **Account Owner** and **System Object** fields because their values cannot be set by the client.
6. If the server returns an empty value in any of the non-compulsory fields, it means that the object's respective property on the server is not set.

Contact

The contact object is used by the client to load and change a contact settings. The client can only **load** contacts that are either created by or assigned to the account that is currently logged on. The client can only **change settings** for the contact that is created by the account that is currently logged on.

Commands

Sender	Command	Command Subtype
Server	LOAD	LOR_CONTACT_OK
Server	OBJ_NEW	LOR_CONTACT_OK
Client	OBJ_SAVE	LOR_CONTACT
Server	OBJ_FIND	LOR_CONTACT_OK

Fields

Position	Contains	Data Type	Required
1	Contact Id	Number	✓
2	Name	String	x
3	Account Owner	String	✓
4	System Object	Boolean	✓
5	Address	String	x
6	Carrier Id	Number	x
7	Notes	String	x

Field Details

- 1 **Address** – pager id, mobile number or email address
 - 1.1 The value depends on the Carrier Id field.

- 1.2 If present, the value must be properly formatted to match the type of address requested by the carrier (e.g. proper pager id, mobile number or email address)
- 2 **Carrier Id** – id of the carrier object as defined on the server.
 - 2.1 The value must be an id of one of the carriers defined on the server.
 - 2.2 Correct values can be looked up on the server.
 - 2.3 The client application can also load carriers' settings and use them to set the field value.

Example:

```
280,Test Save,nicole,0,+64021101212,2,"Testing the "Save" functionality
0D0A Performed through "TCP""
```

Application

1. Fields Address and/or Carrier Id can be empty. This will delete the contact's respective properties on the server.
2. If the server returns an empty value for any of the them, it means that the contact's respective property on the server is not set.

Group:

The group object is used by the client to load and change settings for a group of contacts. The client can only load groups that are either created by or assigned to the account that is currently logged on. The client can only change settings for the group that is created by the account currently logged on.

Commands

Sender	Command	Command Subtype
Server	LOAD	LOR_GROUP_OK
Server	OBJ_NEW	LOR_GROUP_OK
Client	OBJ_SAVE	LOR_GROUP
Server	OBJ_FIND	LOR_GROUP_OK

Fields

Position	Contains	Data Type	Required
1	Group Id	Number	✓
2	Name	String	x
3	Account Owner	String	✓
4	System Object	Boolean	✓
5	Group Contacts	String	x
6	Notes	String	x

Field Details

1. Group Contacts – ids of the contacts that are assigned to the group

- All ids must belong to contacts that are visible to the account
- Ids are separated by the “Field Delimit” character, defined in the Header.

Example:

```
28,Group Office,nicole,0,"59,9,38,80",Office clients
```

Application

1. If the server returns an empty value in the **Group Contacts** field, it means that the group does not have any contact assigned to it.
2. If the **Group Contacts** field contains a value, the client application must parse it on the “Field Delimit” character into tokens.
3. All tokens (contact ids) must be convertible into an integer. If a token cannot be converted into an integer or contains a value equal or less than 'zero', the server will disregard it.
4. When the user adds contacts to the group settings, the client must always return all contacts in the group, not only the ones that are just added. That is because the server always recreates the group contacts from the scratch before saving them.
5. Sending an empty string in the **Group Contacts** field will delete all contacts from that group.

Template:

The template object is used by the client to load and change settings for a template. The client can only load templates that are either created by or assigned to the account that is currently logged on. The client can only change settings for the template that is created by the account currently logged on.

Commands

Sender	Command	Command Subtype
Server	LOAD	LOR_TEMPLATE_OK
Server	OBJ_NEW	LOR_TEMPLATE_OK
Client	OBJ_SAVE	LOR_TEMPLATE
Server	OBJ_FIND	LOR_TEMPLATE_OK

Fields

Position	Contains	Data Type	Required
1	Template Id	Number	✓
2	Name	String	x
3	Account Owner	String	✓
4	System Object	Boolean	✓
5	Template Text	String	x
6	Notes	String	x

Field Details

1. Template Text – actual text of the message template

- One or more lines of free flowing text
- It can contain new line characters (\$ODOA)
- It can contain any other printable characters compatible with the Unicode Basic Multilingual Plane

Example:

```
23,Critical Incident,nicole,0,Critical incident <date> <time> response
has been initiated. Please phone in,Enter current date and time when
using the template
```

Application

1. Field Template Text can be empty. This will delete the template's respective property on the server.
2. If the server returns an empty value for the field, it means that this property is not set.

Schedule

The schedule object is used by the client to load and change a schedule settings. The client can only **load** schedules that are either created by or assigned to the account that is currently logged on. The client can only **change settings** for the schedule that is created by the account that is currently logged on.

Commands

Sender	Command	Command Subtype
Client	OBJ_SAVE	LOR_SCHEDULE
Server	LOAD	LOR_SCHEDULE_OK
Server	OBJ_NEW	LOR_SCHEDULE
Server	OBJ_FIND	LOR_SCHEDULE_OK

Fields

Position	Contains	Data Type	Required
1	Schedule Id	Number	✓
2	Name	String	x
3	Account Owner	String	✓
4	System Object	Boolean	✓
5	Schedule Type	Number	x
6	End Type	Number	x
7	Enabled	Boolean	x
8	No Of Messages	Number	x
9	Send Interval	Number	x
10	Priority	Number	x
11	Age Limit	Number	x
12	Message Limit	Number	x
13	Start Date	Date Time	x
14	End Date	Date Time	x
15	Schedule Contacts	String	x
16	Email Subject	String	x
17	Message Text	String	x
18	Allow Empty Message	Boolean	x
19	No Of Sent Messages	Number	x
20	Sent Date	Date Time	x
21	Notes	String	x

Fields Details

- 1 Schedule Type** – type of the schedule (send once, hourly, daily, weekly, monthly)
For allowed values see “[Schedule Type](#)”
Default: ST_ONCE (0)
- 2 End Type** – Defines the way to end message scheduling (end on a certain date, end after certain number of messages or both)
For allowed values see “[Schedule End Type](#)”
Default: SET_NONE (0)
- 3 Enabled** – Defines whether the scheduling is on or off for that schedule
Default: False
- 4 No Of Messages** – Defines a number of sent messages after which scheduling will be stopped
Default: 1
- 5 Send Interval** – Defines the interval in which the messages will be scheduled
Default: 1
- 6 Priority** – The [priority](#) of the message in the server's message queue
Default: MPR_NORMAL (4)
- 7 Age Limit** – Defines a number of minutes the server will keep an undelivered message before it is discarded
Default: 120 minutes
- 8 Message Limit** – Defines a period of time inside the scheduling interval, when messages should be paused.
Default: 0 (no message limits)
A four byte number is used and then converted directly into a string. The way the number is used depends on the scheduling type:

8.1 Hourly

- In this scheduling type the user can decide to pause the sending of the messages during a certain period of the day
- The number is divided into two words (word = 2 bytes)
- The least significant word contains the start of the pausing period.
- The most significant word contains the end of the pausing period.
- Both numbers are expressed as a number of minutes from the midnight.

1 byte	2 byte	3 byte	4 byte
Start of the pause		End of the pause	

8.2 Daily, monthly and weekly

- In the daily and monthly scheduling type, the user can exclude certain days from the scheduling
- In the weekly scheduling type, the user can decide to send messages every weekly

interval but only on certain days

- The least significant 7 bits of the number are set on for each day of the week when the scheduling is excluded (daily and monthly) or included (weekly)

The schema shows only the least significant byte of the number:

0 bit	1 bit	2 bit	3 bit	4 bit	5 bit	6 bit	7 bit
Mon	Tues	Wed	Thur	Fri	Sat	Sun	N/A

9 Start Date – Defines the start of the scheduling

Default: Now() + 1 day (automatically set for the next day)

9.1 Send Once – Defines the date the message will be sent on

9.2 Hourly, Daily, Weekly, Monthly – Defines the start of the scheduling period

10 Stop Date – Defines the end date of the scheduling

Default: Now() + 15 days (scheduling duration is automatically set for two weeks)

11 Schedule Contacts – The list of contact ids that the scheduled message will be sent to

Default: " - empty string (no contacts set by the system)

11.1 Contact ids, as defined on the server, are converted to a string

11.2 They are separated by the [Field Delimiter](#) character

11.3 The whole string is enclosed in double quotes (it contains a field delimiter char)

12 Email Subject – Used as an email subject but only if the scheduled message is an email message

Default: **i-Page Server message**

13 Message Text – An actual text content of the page, SMS or email message sent to the receiver

Default: : **i-Page Server scheduled message default**

13.1 The default message is used only the field is empty and the next field, "Allow Empty Message" is set to **false**.

14 Allow Empty Message – if this field is set to true, the "Message Text" field does not need to contain any text.

Default: False.

15 No Of Sent Messages – The number of messages that have been sent since the beginning of the scheduling period

Default: 0

16 Sent Date – A date when the last message message was sent

Example:

```
24,Sales Update,nicole,0,2,3,0,73,3,4,120,112,7DD61A102B00,7DD809102B00,
"1,2,3,18",i-Page Server message,Reminder! Regular sales update
due.,0,0,7DD118102B00,Regular sales update 0D0A Send every 3 days
```

Application

- 1 If the "[Schedule Type](#)" field is "Send Once" - ST_ONCE (0) then values for the fields:

- [End Type](#)
- [No Of Messages](#)
- [Send Interval](#)
- [Message Limit](#)
- [Stop Date](#)

are disregarded by the server. They must be present but they can be empty.

- 2 The [End Type](#) field can be SET_NONE (0) only if the schedule type is "Send Once". For all other schedule types it must be set to either SET_DATE (1), SET_MSG_NO (2) or SET_BOTH (3)
- 3 If the [End Type](#) is SET_DATE (1) or SET_BOTH (3), the "[Stop Date](#)" field must have a valid [DateTime](#) value and the date must be latter than the "[Start Date](#)" and the current date.
- 4 If the End Type is SET_MSG_NO (2) or SET_BOTH (3), than the "[No Of Messages](#)" must be greater that 0.
- 5 For every schedule type (except "Send Once") the "[Send Interval](#)" field must be greater than 0.

Report


The report object is used by the client to load and change a report settings. The client can only **load** schedules that are either created by or assigned to the account that is currently logged on. The client can only **change settings** for the report that is created by the account that is currently logged on.

Commands

Sender	Command	Command Subtype
Client	OBJ_SAVE	LOR_REPORT
Server	LOAD	LOR_REPORT_OK
Server	OBJ_NEW	LOR_REPORT
Server	OBJ_FIND	LOR_REPORT_OK

Fields

Position	Contains	Data Type	Required
1	Report Id	Number	✓
2	Name	String	x
3	Account Owner	String	✓
4	System Object	Boolean	✓
5	Date Range Type	Number	x
6	Date From	DateTime	x
7	Date To	DateTime	x
8	Sorted	Boolean	x
9	Sorted Ascending	Boolean	x
10	Sorted On Field	Number	x
11	Message Statuses	Number	x
12	Message Priorities	Number	x
13	Carrier Types	Number	x
14	Address Types	Number	x
15	Carrier Ids	String	x
16	Message Addresses	String	x
17	Account Ids	String	x
18	Text Search Type	Number	x

Position	Contains	Data Type	Required
19	Case Sensitive Search	Boolean	x
20	Partial Match	Boolean	x
21	Search String	String	x
22	Report Colour	Number	x
23	Font Colour	Number	x
24	Report File Fields	Number	x
25	Auto Report	Boolean	x
26	Auto Report Type	Number	x
27	Auto Report Interval	Number	x
28	Auto Report Start Time	Date Time	x
29	Report Email Carrier Id	Number	x
30	Allow Empty Report	Boolean	x
31	Include Report Header	Boolean	x
Next 3 fields repeat for every email assigned to the report			
32	Email Address	String	x 
33	Email Subject	String	x
34	Email Message	String	x
35	Notes	String	x

Fields Details

- 1 **Date Range Type** – Limits the query within certain date and time intervals
For allowable values see [Report Date Range Type](#)
Default: DRT_RANGE (1)
 - 2.1 If the value of the “[Date Range Type](#)” field is DTR_ALL (0) this field is disregarded by the server report engine
 - 2.2 If the value of the “[Date Range Type](#)” field is DTR_RANGE (1) this field contains the start of the date range interval
 - 2.3 If the value of the “[Date Range Type](#)” field is DTR_BEFORE (2) or DTR_AFTER(3) this field contains the date before/after which all messages will be retrieved
- 3 **Date To** – Defines the end of the date and time interval
Default: Now()
 - 3.1 Used only if the value of the “[Date Range Type](#)” field is DTR_RANGE (1). For all other values this field is disregarded by the server report engine
- 4 **Sorted** – If “True” the report is sorted by the server report engine.

Default: False

- 5 Sorted Ascending** – If “True” the report is sorted in the ascending order, otherwise it is sorted in the descending order
Default: False

- 5.1** If the “Sorted” field is “False” this field is disregarded by the server report engine

- 6 Sorted On Field** – Defines a message field the report is sorted on
For allowable values see “[Stored Messages Fields](#)”
Default: MSG_DATE_SENT (12)

- 7 Message Sent Statuses** – Defines all message sent statuses that are used in the report query
Default: 0

- 7.1** Uses 4 least significant bits of the number to set every sent status required and then converts the result into a string

0 bit	1 bit	2 bit	3 bit
MSS_NOT_SENT	MSS_SENT	MSS_ERROR	MSS_SENT_WITH_ERROR

For available sent statuses, see “[Message Sent Status](#)”

- 8 Message Carrier Types** – Defines all carrier types that are used in the report query
Default: 0

- 8.1** Uses 8 least significant bits of the number to set every carrier type required and then converts the result into a string

0 bit	1 bit	2 bit	3 bit	4 bit	5 bit	6 bit	7 bit
TAP Modem	TAP Direct	GSM SMS	Email	Email Pager	Email SMS	TAP IP	SNPP

For available carrier types, see “[Carrier Type](#)”

- 9 Message Address Types** – Defines all message address types that are used in the report query
Default: 0

- 9.1** Uses 4 least significant bits of the number to set every address type required and then converts the result into a string

0 bit	1 bit	2 bit	3 bit
Unknown	Pager Id	Mobile Number	Email Address

For available address types, see “[Address Type](#)”

- 10 Carrier Ids** – Limits the result query to carriers selected by the user

- 10.1** All ids belong to carriers defined on the server
- 10.2** Ids are converted to a string and delimited with the [Field Delimiter](#) character
- 10.3** The result string is enclosed in double quotes

- 11 Message Addresses** – Limits the result query to receivers' addresses selected by the

user

- 11.1** All addresses (if necessary) are converted to strings and delimited with the [Field Delimiter](#) character
 - 11.2** The result string is enclosed in double quotes
- 12 Account Ids** – Limits the result query to sender accounts selected by the user
 - 12.1** All ids belong to accounts defined on the server
 - 12.2** Ids are converted to a string and delimited with the [Field Delimiter](#) character
 - 12.3** The result string is enclosed in double quotes
- 13 Text Search Type** – The result query can be limited by a text found in the message text. This field defines how the search text entered by the user, should be treated in the query. For available values see [“Text Search Type”](#)
Default: TST_ANY_WORD (0)
- 14 Case Sensitive Search** – If set to “True” the search for the text in the message will be case sensitive.
Default: False
- 15 Partial Match** – If set to “False” the query will search only for whole words, not for words as a part of another larger word.
- 16 Search String** – one or more words, entered by the user, that will be searched by the query in the message text, according to criteria from the [“Text Search Type”](#) field.
- 17 Report Colour** – Defines the colour of the report background, when run on the server. The colour is defined in a RGB format.
E.g. - Red: 0x000000FF; Green: 0x0000FF00; Blue: 0x00FF0000
The number is expressed as an integer and converted to a string.
- 18 Font Colour** – Defines the colour of the report font, when run on the server. The colour is defined in a RGB format.
E.g. - Red: 0x000000FF; Green: 0x0000FF00; Blue: 0x00FF0000
The number is expressed as an integer and converted to a string.
- 19 Report File Fields** – Defines stored message fields that will be sent as a query result in the CSV file, attached to the report email.
Default: 40950
Contains: Transaction Id, Account Id, Address, Carrier Id, Carrier Name, Carrier Type, Message Sent Status, Result Code, Result Description, Message Text and Date Sent
 - 19.1** Uses 24 least significant bits to set a message field that will appear in the report file. A position of the bit is defined by its constant in the [“Stored Messages Fields”](#)
 - 19.2** Number is converted into a string.
- 20 Auto Report** – If set to “True” the report will be run automatically
Default: False
- 21 Auto Report Type** – Defines the type of auto report
For allowable values see [“Auto Report type”](#)
Default: ART_DAILY (0)
 - 21.1** If the field [“Auto Report”](#) is “False”, this field is disregarded by the server report engine

- 22 Auto Report Interval** – Defines an interval (a number of days/hours) in which the report will be generated.
Default: 1 (day/hour)
- 22.1** If the field “[Auto Report](#)” is “False”, this field is disregarded by the server report engine
- 23 Auto Report Start Time** – Defines the report start time
- 23.1 Daily Type** – Defines the time when the report will be generated every <interval> days.
- 23.2 Hourly Type** – Defines the time when the engine will start to generate hourly reports
- 23.3** If the field “[Auto Report](#)” is “False”, this field is disregarded by the server report engine
- 24 Report Email Carrier Id** – Defines the id of the email carrier (registered on the server) that the email engine uses to send emails with the report data attached.
- 25 Allow Empty Report** – It is very likely that there will be no users' messages recorded within some of the report's intervals. In that case the report engine does not create a report and does not send any emails to users. To create the report attachment anyway, set this field to True. The created report will have the header with all its basic and filtering settings displayed and a warning that there are no messages in a report interval period.
Default: False
- 26 Include Report Header** – The user can choose not to include the report header in the attached CSV file by setting this field to False.
Default: True
- 26.1 Report Header contains:**
- Report name
 - Report date and time criteria
 - If the report is an Auto report type - Auto report criteria
 - All filtering criteria set by the user
- 27 Report Email** – Contains a records for every email that will be sent with the report result as an attachment
- 27.1** Every email record has following fields:
- Email Address
 - Email Subject
 - Email Message
- 27.2** Fields in the email record are delimited with the “[Field Delimiter](#)” character
- 27.3** Email records are delimited with the [US](#) (\$1F) control character
- 27.4** The whole resulting string with all the email records is enclosed in double quotes

Example:

```
"john@server.com,Subject for John,Message for John1F  
joe@server.com,Subject for Joe,Message for Joe1F"
```

```
nicky@server.com,Subject for Nicky,Message for Nicky"
```

- 27.5 If any of the strings (subject or message text) contains the "[Field Delimiter](#)", it will be enclosed in duplicate double quotes.

Example:

```
"john@server.com,Subject for John,""Message for John, that contains a
field delimiter""1F
joe@server.com,Subject for Joe,Message for Joe1F
nicky@server.com,Subject for Nicky,Message for Nicky"
```

Example – Whole Report Record :

```
24,Office Report,nicole,0,1,7DD115000000,7DD417000000,
1,0,12,10,16,19,14,"1,3,5","1234567,2345678",nicole,0,0,0,Search
string,8454143,255,40950,0,0,1,7D7809110000,4,0,1,
"john@server.com,Subject for John,Message for John1F
joe@server.com,Subject for Joe,Message for Joe1F
nicky@server.com,Subject for Nicky,Message for Nicky",This is just a
note for the report Office Report1E
```

Application

- 1 There are 4 types of the fields in every report record and they are used by the report engine for different purposes. The fields are used to:
 - Define the report record
 - Build the report run time query
 - Format the report
 - Send the result back to the user via email
- 2 Fields used to define the report
 - Report Id, Name, Account Owner, System Object, Notes
 - 2.1 Fields Report Id, Account Owner and System object are compulsory and must have a valid value.
- 3 Fields used to build the report's run time query
 - 3.1 Auto Reports
 - ◇ Auto Report, Auto Report Type, Auto Report Interval, Auto Report Start Time
 - 3.1.1 If the "Auto Report" field is set to False, the rest of the fields from that group are disregarded by the server report engine
 - 3.2 Regular Reports
 - ◇ Date Range Type, Date From, Date To
 - 3.2.1 These fields are used only if the field "Auto Report" from the previous group is set to False, otherwise are disregarded by the server report engine
 - 3.3 All reports
 - ◇ Message Statuses, Message Priorities, Carrier Types, Address Types, Carrier Ids,

Message Addresses, Account Ids, Text Search Type, Case Sensitive Search, Partial Match, Search String

- 3.3.1** Fields from this group are used by all report types
 - 3.3.2** Fields Message Statuses, Message Priorities, Carrier Types and Address Types have limited number of choices. If no choice is set (value = 0) the option is not used in query. The same happens if all choices are set.
 - 3.3.3** If the value of any of the fields Carrier Ids, Message Addresses or Account Ids is empty string, that option is not used in the report query.
 - 3.3.4** If the value of the field Search String is empty string, then fields Text Search Type, Case Sensitive Search and Partial Match match are disregarded by the report engine.
- 4** Fields used to format the report
 - ◇ Sorted, Sorted Ascending, Sorted On Field, Report File Fields, Allow Empty Report, Include Report Header, Report Colour, Font Colour
 - 4.1** If the field Sorted is set to False then fields Sorted Ascending and Sorted On Field are disregarded by the server report engine
- 5** Fields used to send the report result back to the user via email
 - ◇ Report Email Carrier Id, Email Address, Email Subject, Email Message
 - 5.1** Fields Email Address, Email Subject and Email Message are repeated for every email defined by the user

Folder

Note: A “Folder” object is introduced in a version 1.2.

The folder object is used by the client to load and change a folder settings. The client can only **load** folders that are either created by or assigned to the account that is currently logged on. The client can only **change settings** for the folder that is created by the account that is currently logged on.

Commands

Sender	Command	Command Subtype
Server	LOAD	LOR_FOLDER_OK
Server	OBJ_NEW	LOR_FOLDER_OK
Client	OBJ_SAVE	LOR_FOLDER
Server	OBJ_FIND	LOR_FOLDER_OK

Fields

Position	Contains	Data Type	Required
1	Folder Id	Number	✓
2	Name	String	x
3	Account Owner	String	✓
4	System Object	Boolean	✓
5	Contacts	String	x
6	Groups	String	x
7	Notes	String	x

Field Details

- 1 Contacts** – ids of the contacts that are assigned to the folder
 - All ids must belong to contacts that are visible to the account.
 - Ids are separated by the “Field Delimit” character, defined in the Header.
- 2 Groups** – ids of the groups that are assigned to the folder
 - All ids must belong to groups that are visible to the account.
 - Ids are separated by the “Field Delimit” character, defined in the Header.

Example:

```
211,Folder North,nicole,0,"59,21,38,80","11,9,12,8",Northern region
contacts and group of contacts
```

Application

1. If the server returns an empty value in the **Contacts/Groups** field, it means that the folder does not have any contact/group assigned to it.
2. If the **Contacts/ Groups** field contains a value, the client application must parse it on the "Field Delimit" character into tokens.
3. All tokens (contact/group ids) must be convertible into an integer. If a token cannot be converted into an integer or contains a value equal or less than 'zero', the server will disregard it.
4. When the user adds contacts/groups to the folder settings, the client must always return all contacts/groups in the folder, not only the ones that are just added. That is because the server always recreates the folder contacts/groups from the scratch before saving them.
5. Sending an empty string in the **Contacts/ Groups** field will delete all contacts/groups from that folder.

Appendix A.

List Of Commands

Command	Used For
TCP_ERROR	Returning an i-Page Client/Server Protocol error and i-Page Server system error to the client
CONNECT	<ul style="list-style-type: none"> Connecting to the server (client) Reporting Connected/Not connected status to the client Reporting different connection errors to the client Informing the server that the client is ready to communicate Informing the server that the client is disconnecting Informing the client that the server is closing down Informing the client that the connection is disconnected by the administrator
LOGIN	<ul style="list-style-type: none"> Requesting a login from the client Sending login data to the server Reporting a login success/failure to the client Reporting different login errors to the client Changing account password on the server Reporting changing password success/failure to the client
LOAD	<ul style="list-style-type: none"> Loading contacts, groups, templates, schedules, reports, carriers, message settings, account and system data from the server Returning loading results (success/failure) to the client
OBJ_NEW	Creating a new client contact, contact group, template, schedule and report on the server
OBJ_SAVE	Saving the existing client contact, contact group, template and schedule to the server
OBJ_DEL	Deleting the existing client contact, contact group, template, schedule and report on the server
QUERY_RUN	<ul style="list-style-type: none"> Retrieving one or more system-wide contacts, contact groups, templates, schedules and reports from the server, according to the search criteria set by the client. Running query about sent messages, according to the query criteria set by the client. Running a report.

Command	Used For
ACC_CHNG	<ul style="list-style-type: none">• Changing account settings• Changing the account's system-wide object assignment
MSG_SEND'	<ul style="list-style-type: none">• Sending a page, SMS and/or email message to the server• Receiving message sent results on the client• Reporting different message sent and i-Page system errors to the client
SEND_ESCLT	<ul style="list-style-type: none">• Sending an escalated page, SMS and/or email message to the server• Receiving message sent results on the client• Reporting different message sent and i-Page system errors to the client

List Of Command Subtypes

Set of constants returned in the “[Command Subtype](#)” field of the header with the respective command. They either define the command or show the result of a previously sent command of the same type. All constants defined as internal are not used in a client/server communication. Application software (client and server) is free to use them to dispatch messages internally about its status or an error that occurred.

The value of the constant is shown in the same format as it appears in the header string (2 hexadecimal digits).

Communication Type Legend

INT	Internal
C/S	Client → Server
S/C	Server → Client

Connection Subtypes

Used to define the “**CONNECT**” command or its result.

Constant	Value	Type	Description
CR_CONNECTING	00	Notice	INT Used in the client to set its status during a connection process.
CR_CONNECT	01	Request	C/S The client requests a connection to the server.
CR_CONNECT_OK	02	Result	S/C The client is successfully connected to the server.
CR_NOSERVER_COM	03	Result	INT The client cannot find the COM based server on the machine.
CR_NOSERVER_TCP	04	Result	INT The client cannot find the TCP based server.
CR_CONNECTION_ACCEPTED	05	Result	S/C The client is successfully connected to the server.
CR_CONNECTION_EXCESS	06	Result	S/C The server has refused the client's connection request, because the number of connections exceeds the number of clients the server is registered for.
CR_SERVER_DISCONNECT	07	Notice	S/C The server is closing down.

Constant	Value	Type	Description
CR_CLIENT_DISCONNECT	08	Notice	C/S The client is disconnecting from the server.
CR_ADMIN_DISCONNECT	09	Notice	S/C the administrator has closed this client connection.
CR_CONNECT_LOST	0A	Notice	INT Used by the software to dispatch internal messages when the TCP connection is lost.
CR_NOT_CONNECTED	0B	Result	S/C The server has not yet created a session for the client.
CR_CLIENT_READY	0C	Notice	C/S The client is ready to communicate with the server.
CR_INACTIVE_TIMEOUT	0D	Notice	S/C The server has closed this connection because of inactivity

Login Subtypes

Used to define the “**LOGIN**” command or its result.

Constant	Hex	Type	Description
LR_LOGIN	0	Request	C/S The client is sending login information to the server.
LR_LOGIN_REQUEST	1	Request	S/C The server requests login information from the client.
LR_LOGIN_OK	2	Result	S/C The client has successfully passed the authentication procedure.
LR_LOGIN_ERROR	3	Result	S/C The client has failed the authentication procedure. The reason is either incorrect user name or incorrect password (or both)
LR_ACCOUNT_DISABLED	4	Result	S/C The client has failed the authentication procedure. The reason is that the user's account is disabled by the server administrator.
LR_NORIGHTS	5	Result	Internal. Used by the server when the user tries to log on directly to change the server settings. The login fails because the account does not have sufficient rights.
LR_DUPLICATE_LOGIN	6	Result	S/C The client has failed the authentication procedure. The reason is that the user's account is already logged on the server from the same IP address.
LR_CHANGE_LOGIN	7	Request	C/S The client requests a change of the account's password.
LR_CHANGE_LOGIN_OK	8	Result	S/C The account's password was successfully changed.
LR_CHANGE_LOGIN_ERROR	9	Result	S/C the account's password was not successfully changed. Either old or new password was not successfully confirmed.

System Object Subtypes

Used to define all the commands that deal with any of the objects in the i-Page System. These objects are:

1. Contacts
2. Contact Groups
3. Message Templates
4. Schedules
5. Reports
6. User Accounts
7. Carriers – basic
8. Carriers – with summary
9. System Data
10. Message Settings
11. Sent Messages
12. Folders – Version 1.2

It is used with the following commands:

- LOAD
- OBJ_NEW
- OBJ_SAVE
- OBJ_DEL
- QUERY_RUN
- ACC_CHNG

This command is optional. The client does not need to use it to successfully communicate with the server or to send messages to the server. If it is used, the client can request only data it is interested in.

If the server returns any of the “_OK” constants, the main message string will contain a delimited string with all requested data for all the objects. If the server returns any of the “_ERROR” constants the main message will contain a description of the error.

Constant	Hex	Type	Description
LOR_CONTACT	00	Request	C/S The client requests from the server data about all contacts that are created by or assigned to the client's account.

Constant	Hex	Type	Description
LOR_GROUP	03	Request	C/S The client requests from the server data about all contact groups that are created by or assigned to the client's account.
LOR_TEMPLATE	06	Request	C/S The client requests from the server data about all templates that are created by or assigned to the client's account.
LOR_ACCOUNT	09	Request	C/S The client requests from the server data about the client's account.
LOR_SYSTEM	0C	Request	C/S The client requests from the server data about the i-Page system settings
LOR_CARRIER	0F	Request	C/S The client requests from the server basic data about all carriers that are created on the server.
LOR_CARRIER_SUM	12	Request	C/S The client requests from the server data about all carriers that are created on the server, including brief summary.
LOR_SCHEDULE	15	Request	C/S The client requests from the server data about all schedules that are created by the client's account.
LOR_REPORT	18	Request	C/S The client requests from the server data about all reports that are created by or assigned to the client's account.
LOR_MSG_SETUP	1B	Request	C/S The client requests from the server data about message settings for a particular account
LOR_RUN_REPORT	1E	Request	C/S The client requests from the server to run a selected report
LOR_SENT_MSG	21	Request	Request
LOR_FOLDER	24	Request	C/S The client requests from the server data about all folders that are created by or assigned to the client's account.
LOR_CONTACT_OK	01	Result	S/C The server is sending data about contacts.
LOR_GROUP_OK	04	Result	S/C The server is sending data about contact groups.
LOR_TEMPLATE_OK	07	Result	S/C The server is sending data about templates.

Constant	Hex	Type	Description
LOR_ACCOUNT_OK	0A	Result	S/C The server is sending data about the account logged on the client.
LOR_SYSTEM_OK	0D	Result	S/C The server is sending data about i-Page settings.
LOR_CARRIER_OK	10	Result	S/C The server is sending data about carriers.
LOR_CARRIER_SUM_OK	13	Result	S/C The server is sending data about carriers, including their brief summary.
LOR_SCHEDULE_OK	16	Result	S/C The server is sending data about schedules.
LOR_REPORT_OK	19	Result	S/C The server is sending data about defined report templates.
LOR_MSG_SETUP_OK	1C	Result	S/C The server is sending data about message settings defined for the account.
LOR_RUN_REPORT_OK	1F	Result	S/C The server is reporting that the selected report ran successfully on the server
LOR_SENT_MSG_OK	22	Result	S/C The server is sending data about sent messages based on the user's query
LOR_FOLDER_OK	25	Result	S/C The server is sending data about folders.
LOR_CONTACT_ERROR	02	Result	S/C The server is reporting an error that occurred while trying to create and send data about account's contacts.
LOR_GROUP_ERROR	05	Result	S/C The server is reporting an error that occurred while trying to create and send data about account's contact groups.
LOR_TEMPLATE_ERROR	08	Result	S/C The server is reporting an error that occurred while trying to create and send data about account's templates.
LOR_ACCOUNT_ERROR	0B	Result	S/C The server is reporting an error that occurred while trying to create and send data about the account.
LOR_SYSTEM_ERROR	0E	Result	S/C The server is reporting an error that occurred while trying to create and send data about the i-Page settings.
LOR_CARRIER_ERROR	11	Result	S/C The server is reporting an error that occurred while trying to create and send data about carriers.

Constant	Hex	Type	Description
LOR_CARRIER_SUM_ERROR	14	Result	S/C The server is reporting an error that occurred while trying to create and send data about carriers, with their summary.
LOR_SCHEDULE_ERROR	17	Result	S/C The server is reporting an error that occurred while trying to create and send data about account's schedules.
LOR_REPORT_ERROR	1A	Result	S/C The server is reporting an error that occurred while trying to create and send data about account's report templates.
LOR_MSG_SETUP_ERROR	1D	Result	S/C The server is reporting an error that occurred while trying to send data about account's message settings.
LOR_RUN_REPORT_ERROR	20	Result	S/C The server is reporting an error that occurred while trying to run the selected report on the server
LOR_SENT_MSG_ERROR	23	Result	S/C The server is reporting an error that occurred while trying to run the user's query for sent messages
LOR_FOLDER_ERROR	26	Result	S/C The server is reporting an error that occurred while trying to create and send data about account's folders.

Send Message Subtypes

Constant	Value	Type	Description
SMR_MSG_SEND	00	Request	C/S The client is sending a page, SMS or email message to the server.
SMR_MSG_SENT_OK	01	Result	S/C The message is sent to the carrier.
SMR_NO_CARRIER_ID	02	Result	S/C Error sending the message to the carrier. A carrier id is incorrect or missing.
SMR_NO_DEVICE_ID	03	Result	S/C Error sending the message to the carrier. A device id is incorrect or missing.
SMR_CARRIER_DISABLED	04	Result	S/C Error sending the message to the carrier. The carrier is disabled on the server..
SMR_DEVICE_DISABLED	05	Result	S/C Error sending the message to the carrier. The device is disabled. The carrier is disabled on the server..
SMR_NO_ADDRESS	06	Result	S/C Error sending the message to the carrier. The message address (pager id/mobile number/email address) is corrupted or missing.
SMR_ADDRESS_FORMAT	07	Result	S/C Error sending the message to the carrier. The message address is not properly formatted pager id/mobile number/email address.
SMR_MSG_EXPIRED	08	Result	S/C Error sending the message to the carrier. The message time on the server has expired.
SMR_LENGTH_NOFRAGMENT	09	Result	S/C Error sending the message to the carrier. The message is too long and the server settings do not allow fragmenting.
SMR_LENGTH_TRUNC	0A	Result	S/C The message is sent to the carrier with errors. The message was too long and had to be truncated.
SMR_FRAGMENT_EXCESS	0B	Result	S/C Error sending the message to the carrier. The number of message fragments exceeds the number allowed by the server settings.

Constant	Value	Type	Description
SMR_FRAGMENT_TRUNC	0C	Result	S/C The message is sent to the carrier with errors. The number of message fragment was too big and had to be truncated.
SMR_FRAGMENT_EXPIRED	0D	Result	S/C The message is sent to the carrier with errors. The time of some of the message fragments had expired before they could be sent.
SMR_DEVICE_INIT_ERROR	0E	Result	S/C The server could not initialise the device.
SMR_DIALUP_ERROR	0F	Result	S/C An error dialling a TAP carrier number.
SMR_DIAL_SMS_ERROR	10	Result	S/C An error dialling a mobile number.
SMR_NOLOGON_REQUEST	11	Result	S/C TAP carrier did not respond with the login request.
SMR_LOGON_NOT_ACCEPTED	12	Result	S/C TAP carrier does not accept login.
SMR_NOOK_TOSEND	13	Result	S/C TAP carrier did not give a permission to the server to send the message
SMR_SEND_ERROR	14	Result	S/C A carrier has reported an error while trying to send the message.
SMR_CARRIERADDRESS_FORMAT	15	Result	S/C Carrier address is not properly formatted phone number.
SMR_SMSC_ADDRESS	16	Result	S/C An error setting the SMS Centre address.
SMR_SMS_MODE_ERROR	17	Result	S/C Error setting a SMS mode type.
SMR_SYSTEM_ERROR	18	Result	S/C i-Page Server system error has been reported. [an error specifics, if available]
SMR_IPP_PROTOCOL_ERROR	19	Result	S/C i-Page Protocol error has been reported. [an error specifics, if available]

List Of Control Characters

The following is the list of ASCII control characters that have a special functionality in the iPP protocol:

Constant	Name	Decimal	Hex	Usage
Header				
US	Date Delimiter (<i>unit separator</i>)	#31	\$1F	<ul style="list-style-type: none"> • Delimits dates in the variable part of the header • Marks the end of the Date Received part • Marks the start of the Date Sent part
GS	Client Data Delimiter (<i>group separator</i>)	#29	\$1D	<ul style="list-style-type: none"> • Marks the end of the Date Sent part • Marks the start of the Client Data part
STX	Header Delimiter (<i>start of text</i>)	#2	\$2	<ul style="list-style-type: none"> • Marks the end of the Client Data part • Delimits the Header from the Main Message part
Main Message				
ETX	Message Text Delimiter (<i>end of text</i>)	#3	\$3	<ul style="list-style-type: none"> • Used only with the "Send Message" command • Marks the end of the user's message text and the start of the message data in the "SMR_MSG_SEND" subtype • Marks the end of the send result description and the start of the message data in all other "SMR_*" subtypes
FD	Field Delimiter ¹	#44	\$2C	<ul style="list-style-type: none"> • Delimits multiple fields
RS	Record Separator	#30	\$1E	<ul style="list-style-type: none"> • Delimits records of data

¹ IPP protocol default – comma character (',')

Appendix B.

Constants

Message Priorities

Priority	Constants	Values
Critical	MPR_CRITICAL	1
Urgent	MPR_URGENT	2
High	MPR_HIGH	3
Normal	MPR_NORMAL	4
Lower	MPR_LOWER	5
Low	MPR_LOW	6

Note: Messages with a lower number have a higher priority.

Default: Normal.

Message Affix Types

Affix Types	Constant	Value	Explanation
None	MAT_NONE	0	No affix is used
Prefix	MAT_PREFIX	1	Text/Date added as a prefix
Suffix	MAT_SUFFIX	2	Text/Date added as a suffix
Both	MAT_BOTH	3	Text/Date added as a prefix and as a suffix

Note: Used to add a constant text and/or date time stamp to every message.

Default: MAT_NONE

Address Type

Address Type	Constant	Value
Unknown	AT_UNKNOWN	0
Pager Id	AT_PAGER	1
Mobile Number	AT_SMS	2
Email Address	AT_EMAIL	3

Carrier Type

Carrier Type	Constant	Value
Unknown	CTT_UNKNOWN	-1
Tap Modem	CTT_TAP_MODEM	0
Tap Direct	CTT_TAP_DIRECT	1
SMS	CTT_GSM_SMS	2
Email	CTT_EMAIL	3
Email Pager	CTT_EMAIL_PAGER	4
Email SMS	CTT_EMAIL_SMS	5
TAP IP	CTT_TAP_IP	6
SNPP	CTT_SNPP	7

Text Encoding Type

Type Value	Description
0	Plain text
1	The text is encoded as a Unicode string with 16-bit characters
2	The text is encoded as an ANSI string with 8-bit characters

Sent Messages Query Type

Query Types	Constant	Value	Explanation
All	SMT_ALL	0	Show all messages
Last <#>	SMT_LAST	1	Show last <number> messages
Today	SMT_TODAY	2	Show messages sent today
On Date	SMT_DATE	3	Show messages sent on <date>
Before Date	SMT_BEFORE	4	Show messages sent before <date>
After Date	SMT_AFTER	5	Show messages sent after <date>
Between Dates	SMT_BETWEEN	6	Show messages sent from <from_date> <to_date>

Find Object Query Type

Type Value	Description
0	Find all objects
1	Find all objects which name starts with the "search string"
2	Find all objects which name contains the "search string"

Contact Display Type

Deprecated – in version 1.2 replaced with [Tree View Display Type](#)

Contact Display Type	Constant	Value
Contacts and Groups	CDT_BOTH	0
Contacts Only	CDT_CONTACTS	1
Groups Only	CDT_GROUPS	2

Note: Used to set the way the contacts and contact groups are displayed.

Default: CDT_BOTH

Tree View Display Type

Object	Folder			Group			Contact		
	Display	Position		Display	Position		Display	Position	
Bit	8	7	6	5	4	3	2	1	0
Default	0	1	1	1	1	0	1	0	1

Note: Data for every object are stored in 3 bits. First 2 bits store its position - 1st (01), 2nd (10) or 3rd (11) If the object's 3rd bit is set then the object is visible.

Default: 245 – Contacts first (displayed), groups second (displayed), folders third (NOT displayed – (MSB) 011 110 101 (LSB)

Message Delete Type

Message Delete Type	Constant	Value
All Messages	MDT_ALL	0
Old Messages	MDT_OLD	1
Selected Message	MDT_SELECTED	2

Note: Used to set the type of delete operation that will be performed on stored messages.

Default: MDT_BOTH

Schedule Type

Constant	Value	Description
ST_ONCE	0	Message is scheduled only once
ST_HOURLY	1	Message is scheduled in hourly intervals
ST_DAILY	2	Message is scheduled in daily intervals
ST_WEEKLY	3	Message is scheduled in weekly intervals
ST_MONTHLY	4	Message is scheduled in monthly intervals

Default: ST_ONCE

Schedule End Type

Constant	Value	Description
SET_NONE	0	No scheduling end defined
SET_DATE	1	Scheduling ends on the set date
SET_MSG_NO	2	Scheduling ends after number of messages has been sent
SET_BOTH	3	Scheduling ends on the set date or after a number of messages has been sent – whichever comes first

Default: SET_NONE

Report Date Range Type

Constant	Value	Explanation
DRT_ALL	0	Retrieve all messages
DRT_RANGE	1	Retrieve messages sent within a date range
DRT_BEFORE	2	Retrieve messages sent before a certain date
DRT_AFTER	3	Retrieve messages sent after a certain date

Stored Messages Fields

Constants	Value	Contains
MSG_SYS_ID	0	Message Id (system-wide unique)
MSG_SENT_ID	1	Transaction Id (assigned by a sender)
MSG_ACCOUNT	2	Account Id
MSG_CLIENT_IP	3	Client IP
MSG_ADDRESS	4	Message Address
MSG_CARRIER_ID	5	Carrier Id

Constants	Value	Contains
MSG_CARRIER_NAME	6	Carrier Name
MSG_CARRIER_TYPE	7	Carrier Type
MSG_STATUS	8	Sent Status
MSG_RESULT_CODE	9	Sent Result Code
MSG_RESULT_DESC	10	Sent Result Description
MSG_TEXT	11	Message Text
MSG_DATE_SENT	12	Date Sent
MSG_DATE_RECEIVED	13	Date Received
MSG_PRIORITY	14	Message Priority
MSG_PRIORITY_STR	15	Message Priority Text
MSG_ADDRESS_TYPE	16	Address Type
MSG_CONNECTION	17	Connection Type
MSG_FRAG_NO	18	Number of Fragments
MSG_FRAG_SENT	19	Number of Sent Fragments
MSG_FRAG	20	Fragmented
MSG_SCH	21	Scheduled
MSG_ESC	22	Escalated
MSG_ACK	23	Acknowledged

Message Sent Status

Constant	Value	Explanation
MSS_NOT_SENT	0	Message not sent
MSS_SENT	1	Message sent
MSS_ERROR	2	Message not sent because of error
MSS_SENT_WITH_ERROR	3	Message sent with error

Text Search Type

Constant	Value	Explanation
TST_ANY_WORD	0	Find any word in message
TST_ALL_WORDS	1	Find all words in message (not in the same order)
TST_EXACT_PHRASE	2	Find exact phrase in message

Auto Report type

Constant	Value	Explanation
ART_DAILY	0	Runs every <interval> number of days
ART_HOURLY	1	Runs every <interval> number of HOURS

Appendix C.

The whole message that is sent to the server must be converted into one of the following formats:

- ASCII – only characters from the basic 7-bit ASCII encoding set can be used
- ANSI – all characters from the extended 8-bit ASCII set. The server will use the extended set that is defined as its current computer's code page
- UTF-8
- UTF-16 LE (little-endian byte order)
- UTF-16 BE (big-endian byte order)

The same format must be set on the server. The message that is returned from the server is always in the encoding form selected by the user on the server. The server recognises only characters from the Basic Multilingual Plane codespace.

Data Types

String

The definition of the different types of strings in the iPP protocol is not based on the size of their basic character data type (7, 8, 16 or 32 bits) but on the range of the codespace they can be mapped to. A client application is free to use internally any type of a string as long as it can be converted into the form selected above.

ASCII String

Uses basic 7-bit ASCII character set in the range #0..#127 (\$0..\$7F).

The protocol also allows usage of the extended 8-bit ASCII character set, because all messages at the end must be converted into the UTF-8 string. i-Page server application is a Unicode (UTF-16) based software and does not depend on the local machine code page. If the client application uses extended ASCII, developers must be aware that their clients could run on machines with a different code page set.

Unicode String

Uses any character that corresponds to the Unicode character from the Basic Multilingual Plane codespace.

Boolean

A Boolean type is converted into a string with the following values:

true	'1'
false	'0'

Number

A decimal number of any size converted directly into a string.

Example:

0 → '0'
 123 → '123'
 34567 → '34567'
 256897 → '256897'

Integer

32–bits signed number.

An integer type is converted into a hexadecimal string that occupies 8 characters. If the resulting string is less than 8 hexadecimal characters long, it is left padded with the '0' (\$30 – #48) character.

Not case sensitive, but i-Page Server always returns hexadecimal characters in upper case.

Examples:

0 – '00000000'
 123 – '0000007B'
 3,456 – '00000D80'
 456,987 – '0006F91B'

DateTime

1. A DateTime value is broken into: days, months, years, hours, minutes and seconds.
2. Each of these values is converted into a hexadecimal number, as follows:
 - Year 3 hexadecimals
 - Month 1 hexadecimal
 - Day 2 hexadecimals
 - Hour 2 hexadecimals
 - Minutes 2 hexadecimals
 - Seconds 2 hexadecimals
3. Each hexadecimal number is left-padded with '0' (\$30 – #48) if necessary.
4. All hexadecimals are concatenated into a string in the following order:

year→**month**→**day**→**hour**→**minutes**→**seconds**

Examples:

DateTime	Interim Value	Final Value
29/09/2010 12:32:45	7DA 9 1D 0C 20 2D	7DA91D0C202D
30/10/2010 15:45:08	7DA A 1E 0F 2D 08	7DAA1E0F2D08
28/12/2011 23:59:11	7DB C 1C 17 3B 0B	7DBC1C173B0B

IP Address

- An address is broken down into 4 strings, parsed on the dot '.' character (\$2E – #46).

Example:

10.58.2.199

10 – 58 – 2 – 199

- Every string is converted into a number.
- The resulting number is converted into two hexadecimal characters.

Example:

A – 3A – 2 – C7

- If the resulting string is less than two hexadecimal character, it is left padded with the '0' (\$30 – #48)

Example:

0A – 3A – 02 – C7

- all resulting strings are then concatenated in the left-to-right order.

Example:

0A3A02C7

Examples:

IP Address	Octets				Hex				Result
10.58.2.199	10	58	2	199	0A	3A	02	C7	0A3A02C7
192.168.0.1	192	168	0	1	C0	A8	00	01	C0A80001
69.89.22.105	69	89	22	105	45	59	16	69	45591669
74.220.195.50	74	220	195	50	4A	DC	C3	32	4ADCC332
66.163.168.210	66	163	168	210	42	A3	A8	D2	42A3A8D2

Appendix D.

TCP/IP Byte Order

When you are writing or reading a numerical value directly to or from the socket, you must be aware of its byte order. TCP/IP protocol defines big-endian as the standard network byte order for all numeric values. That means that the most significant byte is sent first. On the other hand, all Intel based processors use little-endian as their byte order, which means that the least significant byte is at the lowest address. The other bytes follow in increasing order of significance.

On Intel platforms it is usually necessary to convert from host byte order (little-endian) to TCP/IP network byte order (big-endian) and the other way around. This can be done by swapping bytes position.

The only time when you need to do in i-Page Client/Server Protocol is when you are writing or reading the [message size](#).

The easiest way to do it is to use functions from the WinSock library.

Host To Network

Normal Message Size

The `htons` (`host_to_network_short`) function takes a 16-bit number in host byte order and returns a 16-bit number in network byte order used in TCP/IP networks.

```
u_short htons(u_short hostshort);  
//hostshort is a 16-bit number in host byte order.
```

Large Message Size

The `htonl` (`host_to_network_long`) function takes a 32-bit number in host byte order and returns a 32-bit number in network byte order used in TCP/IP networks.

```
u_long htonl(u_long hostslong);  
//hostslong is a 32-bit number in host byte order.
```

Network To Host

Normal Message Size

The `ntohs` (`network_to_host_short`) function takes a 16-bit number in TCP/IP network byte order and returns a 16-bit number in host byte order.

```
u_short ntohs(u_short netshort);  
//netshort is a 16-bit number in TCP/IP network byte order.
```

Large Message Size

The `ntohl` (`network_to_host_long`) function takes a 32-bit number in TCP/IP network byte order and returns a 32-bit number in host byte order.

```
u_long ntohs(u_long netlong) ;
//netlong is a 32-bit number in TCP/IP network byte order.
```

Note: To use these functions, you do not need to initiate (WSPStartup) or close (WSPCleanup) the WinSock library.

Examples:

Normal Message Size

Decimal		Hexadecimal	
<i>Little-endian</i>	<i>Big-endian</i>	<i>Little-endian</i>	<i>Big-endian</i>
118	30208	00 76	76 00
256	1	01 00	00 01
450	49665	01 C2	C2 01
1,123	25348	04 63	63 04

Large Message Size

Decimal		Hexadecimal	
<i>Little-endian</i>	<i>Big-endian</i>	<i>Little-endian</i>	<i>Big-endian</i>
118	1,979,711,488	00 00 00 76	76 00 00 00
256	65,536	00 00 01 00	00 01 00 00
450	3,254,845,440	00 00 01 C2	C2 01 00 00
1,123	1,661,206,528	00 00 04 63	63 04 00 00

Delimiting Strings

In i-Page Protocol strings can be delimited by the “[Field Delimiter](#)” character, defined in the [Header](#) by the message sender. Delimited strings can be used only in the [Main Message](#) field.

In all other fields where strings are used, they are byte counted.

In some cases delimited strings must be enclosed in double quotes ([\\$22](#) – [#34](#)):

1. When the string contains the “[Field Delimiter](#)” character
2. When one or more words in the string are already enclosed in double quotes

In that case all double quotes that are already in the string must be duplicated.

Example:

Hello world from WiPath and 'kiwi' programmers²

“Hello, world from WiPath and 'kiwi' programmers”³

“Hello world from “”WiPath”” and 'kiwi' programmers”⁴

Warning: *Never enclose strings in other fields in double quotes.*

² The string does not need to be enclosed in double quotes if single quotes are used, like in the word 'kiwi'

³ The string is enclosed in double quotes because it contains a “[Field delimiter](#)” character, defined as COMMA (,).

⁴ The string is enclosed in double quotes, because it already contains a quoted string “WiPath” - in that case double quotes are duplicated

Appendix E.

Message Sequence Example

- Different message fields are coloured differently
- The “Message Command” field and all delimiters are coloured in bright red
 - Delimiters used:
 - ◇ Date delimiter – US (unit separator) – **\$1F**
 - ◇ Client data delimiter – GS (group separator) – **\$1D**
 - ◇ Header delimiter – STX (start of text) – **\$2**
 - ◇ Message separator – ETX (end of text) – **\$3**
 - ◇ Message record separator – RS (record separator) – **\$1E**
 - ◇ Field separator – defined here as a COMMA ','

Client	Server
12...CONNECT01000020B4,7DD6060A35371F76BC1E 0000001DUnknown2Connecting to i-Page Server on 10.58.2.199:6022	
	11....LOGIN01000020DA,7DD6060E2F121F7DD606 0E2F121DUnknown2Connection not accepted by i-Page ServerDAThe server requests login
11....LOGIN0000020B5,7DD6060B0D141F76BC1E 0000001Dnicole2nicole,7F7141132F905A74836EA 2C53711904F,1	
	11...CONNECT0B000020DC,7DD6060E34041F7DD606 0E34041Dnicole2Not connected to i-Page Server on 10.58.2.199:6022
	11...CONNECT05000020DC,7DD6060E34041F7DD606 0E34041Dnicole2Connection accepted by i- Page ServerDAConnection no 1 of 11
	11....LOGIN02000020DC,7DD6060E34051F7DD606 0E34051Dnicole2Account "nicole" successfully logged on i-Page Server
11...CONNECT0C000020D9,7DD6060C0D2D1F76BC1E 0000001Dnicole2Client "nicole" on 10.58.2.199 ready to communicate	

Client	Server
<pre>11..MSG_SEND000000000,7DD60C0D32131F76BC1E 0000001Dnicole2 Test message from Denny to the SMTP carrier3622,4,joe@server.com,4,194,i-Page Multi Message1E Test message from Georgeto the SMS carrier3623,2,0273214569,4,194,i-Page Multi Message1E Hello, world from Sharon and her 'good' "friends"3624,1,2643029,4,194,i-Page Multi Message1E Test message from Harry to the TAP Direct carrier3625,3,1234568,4,194,i-Page Multi Message</pre>	
	<pre>11..MSG_SEND010000026E,7DD60C0D34301F7DD60C 0D34301Dnicole2Message sent OK. 3joe@server.com,3,Test message from Denny to the SMTP carrier</pre>
	<pre>11..MSG_SEND050000026F,7DD60C0D34301F7DD60C 0D34301Dnicole2Device "GSM Modem" is disabled.DAContact your administrator. 30273214569,2,Test message from Georgeto the SMS carrier</pre>
	<pre>11..MSG_SEND0100000270,7DD60C0D34301F7DD60C 0D353B1Dnicole2Message sent OK. 32643029,1,"Hello, world from Sharon and her 'good' ""friends""</pre>
	<pre>11..MSG_SEND1100000271,7DD60C0D34301F7DD60C 0D35261Dnicole2No login request from the carrier.31234568,1,Test message from Harry to the TAP Direct carrier</pre>
<pre>11...CONNECT0800002133,7DD60C0E0F3B1F76BC1E 0000001Dnicole2Account "nicole" logging out from i-Page Server</pre>	