INTER-ROLLER BAGGAGE HANDLING SYSTEM
HIGH LEVEL CONTROL SYSTEM

INTERFACE SPECIFICATION
TRANSPORT PROTOCOL: RFC1006

Document No.:	IR-102-05
Project No.:	IR-BHS System
Author:	XJ
Release Date:	16-Aug-2005
Revision:	1.00
Pages:	10

© Inter-Roller Engineering Limited

CONTENTS

© Inter-Roller Engineering Limited

1. REVISIONS

Rev.	Issues	Date	Init.	Description
1	00	16-Aug-2005	XJ	Initial version.

© Inter-Roller Engineering Limited

2. PREFACE

2.1 DOCUMENT OBJECTIVE

This document specifies and expands on the technical details of the RFC 1006 transport protocol in IR-BHS, left over from [IR-102-04-1.00 IS_TP_FrameOnRFC1006.doc].

2.2 SCOPE

It is the scope of this document to state all necessary information needed to comprehend the technical details of the RFC 1006 protocol.

2.3 AUDIENCE

This interface specification is intended as a technical document specifying the protocols. Also the document serves, as the technical basis when deciding which standard transport protocols should be used in a particular solution.

2.4 DOCUMENT ORGANISATION

This Interface Specification contains the following chapters:
Chapter 3, "INTRODUCTION"
This chapter supplies an overview of the Frame protocol stack, introducing the layers.
Chapter 4, "RFC 1006 Details"
This chapter specifies RFC 1006 protocol in greater detail.
Chapter 5, "REFERENCES"
This chapter contains a list of the references used in the specification.

2.5 DOCUMENT LIMITATIONS

2.6 DOCUMENT MAINTENANCE

This document is the one of the IR project document suites and maintained by Inter-Roller Engineering Limited.

© Inter-Roller Engineering Limited

3. INTRODUCTION

3.1 OVERVIEW

The RFC1006 protocol is a protocol layer in the IR-BHS RFC 1006 transport protocol stack, which consists of a number of logical layers on top of Ethernet. The reader is referred to [IS -RFC-1006] for further detail on the protocol stack.

RFC 1006 is a message-based protocol, where a message consists of a header and a data block.

The IR-BHS implementation of the RFC 1006 protocol does not a fully implement the RFC 1006 protocol as specified in the public RFC 1006 protocol specification, refer to [RFC-1006]. Rather the focus has been to create a simple and reliable communication protocol. This implementation is primarily developed to meet the requirements from the Siemens SIMATIC PLC. The RFC 1006 protocol is designated ISOonTCP in a Siemens PLC. The reader should be aware that the protocol specified in this document is implemented fully by the network card of a Siemens PLC, i.e. no user program necessary to implement this protocol.

© Inter-Roller Engineering Limited

4. RFC 1006 DETAILS

4.1 INTRODUCTION

The RFC1006 protocol is a message-based protocol. A message consists of a header and a data block.

The header that is referred to as the TPKT-header is always 4 bytes long, and consist of 2 fixed bytes (version + one unused byte) and a two-byte length field specifying the length of the entire telegram including the TPKT-header itself.

	Version	Not Used	Length	
Byte	**0**	**1**	**2**	**3**
Hex	03	00	xx	xx

After this main header an extra header is included specifying (among other things) the type of the telegram. Three telegrams types are implemented in the RFC1006 protocol.

- Connection Request (CR)
- Connection Confirm (CC)
- Data Transfer (DT)

4.2 CONNECTION ESTABLISHMENT

The first two telegrams are used during connection establishment to exchange various identifiers and configuration parameters.

Before a connection can be used to transfer data the two partners must exchange synchronization telegrams. One of the partners must send the '**Connection Request**' telegram and the other must receive this and respond with a '**Connection Confirm**' telegram. If one of the partners receives data in the CR or CC telegram that are illegal or unexpected the receiving part must close down the connection.

When the CR and CC telegrams has been exchanged and accepted by both partners the connection can be used for transferring data, using the data telegram.

4.2.1 Connection Request

Before an RFC1006 connection can be used for data transfer the partner must exchange various ID's and configuration parameters. One part (and only one - normally the client part) will send a Connection Request telegram, and expect to receive a Connection Confirm telegram in response.

A CR telegram could look like this:

	TPKT-Header				CR TPDU										
	VR		Length		LI	CO	DST		SRC		CL	TY	LG	VA	TY
Byte	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
Hex	03	00	00	1d	18	e0	00	00	00	01	00	**c0**	01	0b	**c1**

© Inter-Roller Engineering Limited

	CR TPDU (continued)													
	LG	VA					TY	LG	VA					
Byte	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Hex	05	4c 'L'	4f 'O'	43 'C'	41 'A'	4c 'L'	**c2**	06	52 'R'	45 'E'	4d 'M'	4f 'O'	54 'T'	45 'E'

The TPKT header shown in the example is specifying a total length of the telegram to **29** bytes. The Connection Request part (CR TPDU) consist of the following fields.

- **LI**: The length of the TPDU field excluding itself (total length of telegram - 5)
- **CO**: TPDU code. Always 0xe0 for CR
- **DST**: Destination reference. Always set to 0x00 in the CR
- **SRC**: Source reference. Must a number > 0.
- **CL**: Protocol class. Always set to 0x00.

The following data is called the **C-parameters**. They all consist of three fields (TY=type, LG=length and VA=value).

- **c0**: TPDU size. This value specifies the maximum length that is allowed for the TPDU part of the telegrams. The maximum size in bytes is calculated as 2^value. E.g. a TPDU size of 10 specifies the maximum number of bytes to 1024.
- **c1**: Calling TSAP. This specifies a 1-8 character long ID that both ends must agree on. In the example the calling TSAP was set to 'LOCAL'.
- **c2**: Called TSAP. This specifies a 1-8 character long ID that both ends must agree on. In the example the called TSAP was set to 'REMOTE'.

After sending the CR telegram to RFC1006 protocol handler will wait for a CC telegram.

4.2.2 Connection Confirm

The 'connect confirm' (CC) telegram uses the same fields as the CR telegram. Some of the fields will contain different values these are explained below. An example of a CC telegram is shown here:

	TPKT-Header				CC TPDU										
	VR		Length		LI	CO	DST		SRC		CL	TY	LG	VA	TY
Byte	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Hex	03	00	00	1d	18	d0	00	01	00	01	00	**c0**	01	0b	**c1**

	CR TPDU (continued)													
	LG	VA						TY	LG	VA				
Byte	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Hex	06	52 'R'	45 'E'	4d 'M'	4f 'O'	54 'T'	45 'E'	**c2**	05	4c 'L'	4f 'O'	43 'C'	41 'A'	4c 'L'

The CC telegram has the following differences compared to the CR telegram.

© Inter-Roller Engineering Limited

- **CO**: TPDU code. Always **0xd0** for the CC telegram.
- **DST**: Destination reference. Must contain the value received in the 'SRC'-field from the CR telegram.

The 'Calling TSAP' and 'Called TSAP' fields are exchanged in the C-parameters fields, so that the 'Calling TSAP' from the Connection Request telegram is returned in the 'Called TSAP' field and visa versa.

4.3 DATA TRANSFER

When the CC/CR telegrams have been exchanged, the connection is ready to be used for data transfer. The telegram format could look like the following, if 4 bytes of data are to be transferred.

	TPKT-Header				Data TPDU						
	VR		Length		LI	CO	ET	Data			
Byte	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Hex	03	00	00	0a	02	f0	80 or 00	d1	d2	d3	d4

TPDU Fields:

- **LI**: For data transfer telegrams the LI field is always set to **0x02**.
- **CO**: TPDU code. Always set to **0xf0** data telegrams.
- **ET**: End-Of-Transfer indication. If this is the end of a data transfer this field is set to **0x80**. If a telegram is split into multiple telegrams the first telegrams have this field set to **0x00**, and the last has 0x80 (see below).

If the length of the data to be transferred is larger than allowed according to the Maximum TPDU size, the data must be split into multiple telegrams. The 'ET' field is set to 0x00 on all the telegrams except the last where 'ET' is set to 0x80. The following example shows a telegram that has been split into three telegrams.

Telegram 1 (ET = 0x00)

	TPKT-Header				Data TPDU						
	VR		Length		LI	CO	ET	Data			
Byte	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**.**	**M**
Hex	03	00	xx	xx	02	f0	00	d1	d2	.	dm

Telegram 2 (ET = 0x00)

	TPKT-Header				Data TPDU						
	VR		Length		LI	CO	ET	Data			
Byte	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**.**	**M**
Hex	03	00	xx	xx	02	f0	00	dm+1	dm+2	.	2*dm

© Inter-Roller Engineering Limited

Telegram 3 (ET = 0x80)

	TPKT-Header				Data TPDU						
	VR		Length		LI	CO	ET	Data			
Byte	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**.**	**.**	**NN**
Hex	03	00	xx	xx	02	f0	80	(2*dm)+1	.	.	dEnd

© Inter-Roller Engineering Limited

5. REFERENCES

Abbreviation	Reference
[RFC-1006]	RFC 1006 – ISO Transport Service on top of the TCP. Version 3. http://www.faqs.org/rfcs/rfc1006.html
[IR-102-04-1.00 IS_TP_FrameOnRFC1006.doc]	Interface Specification, Transport Protocol: FrameOnRFC1006 Inter-Roller document ID: IR-102-04

© Inter-Roller Engineering Limited