MCPC TR-001 Ver.1.0

BASIC IMAGING PROFILE

Technical References

Ver.1.0

February 20, 2004

Mobile Computing Promotion Consortium

(MCPC)

Revision History

Revision	Date	Section	Comments
1.0	2004/02/20		Ver1.0

Contents

Abstract

This document is intended to supplement Basic Imaging Profile (BIP) specifications by Bluetooth Special Interest Group (SIG), and to improve interoperability between BIP compliant devices. This Q&A format document illustrates recommendable implementation methods and examples.

Warning

This document is neither mandatory requirements nor impairments to the rights of Bluetooth. Suggested uses in this document shall be adopted by users at their sole risk and responsibilities. MCPC shall make any explicit or implicit guarantee or warranty including but not limited to legal infringements, marketability and consistency to certain purposes.

Reference Standards and Specifications

Shown below is a table of standards and specification documents referred by this document. The latest versions are used unless otherwise stated.

References
[1] BASIC IMAGING PROFILE Interoperability Specification
[2] IMPLEMENTATION GUIDELINES BASIC IMAGING PROFILE, Revision 1.0
[3] JEITA Standard Exchangeable image file format for digital still camera, Version 2.21
[4] JEITA Standard Design rule for Camera File system, Version 2.0
[5] ISO/IEC 10918-1 / ITU-T Recommendation T.81 information technology - Digital compression and coding of continuous-tone still images - Requirements and guide-lines
[6] IrOBEX Specification, Version 1.2, Infrared Data Association
[7] Specification of the Bluetooth System, Assigned Numbers Specification

Definition of abbreviations and symbols

Shown below is a table of abbreviations and symbols used in this document.

Abbreviation/Symbol	Definition
CoD	Class of Device
DCF	Design rules for Camera File system
OBEX	Object Exchange Protocol
SDP	Service Discovery Protocol
Exif	Exchangeable image file for digital still cameras
XML	Extensible Markup Language
DTD	Document Type Definition

Q1. Preferred-format in capabilities object

A.

A preferred-format element is used by the Imaging Responder to inform an Imaging Initiator of the most appropriate image format among acceptable formats. However, it is valid only when the format of a native image is identical to a specified image format or when an Imaging Initiator can convert a transmission image to a specified image format. The image format specified in this preferred-format shall be subset of image-formats described later in Q2. And the Imaging Responder, who wishes to receive a native image without conversion of the image formats, shall not specify a preferred-format element.

As the number of appearances of preferred-format elements is specified as “?” in DTD of imaging-capabilities object, multiple descriptions are not possible and if unnecessary, it may be omitted. And according to an attribute-list declaration of preferred-format element, four attribute values, “encoding”, “pixel”, “transformation” and “maxsize” are defined. Among them, “encoding” is a mandatory attribute, which shall be described upon describing a preferred-format element. On the other hand, “pixel” may be omitted. However it is recommended to describe it as much as possible in order to enhance interoperability. Although “pixel” may be described either in a fixed value (e.g. ”640*480”) or a range of pixel sizes (e.g. ”1*1-1600*1200”), descriptions in a fixed value is recommended.

Shown below is an element type (ELEMENT) declaration and an attribute-list (ATTLIST) declaration related to a preferred-format element extracted from DTD of imaging-capabilities object.

```
<!ELEMENT preferred-format EMPTY>
<!ATTLIST preferred-format
        encoding CDATA #REQUIRED
        pixel CDATA #IMPLIED
        transformation NMTOKENS #IMPLIED ”stretch crop fill”
        maxsize CDATA #IMPLIED>
```

Description examples of a preferred-format element are shown as:

- Example1

  The Imaging Responder is able to receive JPEG encoded images and BMP encoded images of any pixel sizes but prefer JPEG encoded image with size 320*240 pixels.

  ```
  <imaging-capabilities version=”1.0”>
  <preferred-format encoding=”JPEG” pixel=”320*240”/>
  <image-formats encoding=”JPEG” pixel=”1*1-65535*65535”/>
  <image-formats encoding=”BMP” pixel=”1*1-65535*65535”/>
  </imaging-capabilities>
  ```

- Example2

  The Imaging Responder is able to receive JPEG encoded images of 160*120 pixel size and 640*480 pixel size but prefer JPEG encoded images of 640*480 pixel size.

  <imaging-capabilities version=”1.0”>

  <preferred-format encoding=”JPEG” pixel=”640*480”/>

  <image-formats encoding=”JPEG” pixel=”160*120”/>

  <image-formats encoding=”JPEG” pixel=”640*480”/>

  </imaging-capabilities>

- Example3

  The Imaging Responder is able to receive JPEG encoded images of any pixel sizes, but prefer JPEG encoded images of 640*480 (VGA) ~ 1024*768 (XGA) pixel size.

  <imaging-capabilities version=”1.0”>

  <preferred-format encoding=”JPEG” pixel=”640*480-1024*768”/>

  <image-formats encoding=”JPEG” pixel=”1*1-65535*65535”/>

  </imaging-capabilities>

- Example4

  The Imaging Responder is able to receive JPEG encoded images of all pixel sizes, but it prioritizes the reception of JPEG encoded images whose file size is 50000 bytes or smaller.

  <imaging-capabilities version=”1.0”>

  <preferred-format encoding=”JPEG” maxsize=”50000”/>

  <image-formats encoding=”JPEG” pixel=”1*1-65535*65535”/>

  </imaging-capabilities>

- Example5

  The Imaging Responder is able to receive JPEG encoded images and BMP encoded images of all pixel size, but it prioritizes the reception of JPEG encoded images.

  <imaging-capabilities version=”1.0”>

  <preferred-format encoding=”JPEG”/>

  <image-formats encoding=”JPEG” pixel=”1*1-65535*65535”/>

  <image-formats encoding=”BMP” pixel=”1*1-65535*65535”/>

  </imaging-capabilities>

As shown in above examples, when describing a preferred-format element, “pixel” may be described either in a fixed value (e.g. ”640*480”) or a range of pixel sizes (e.g. ”1*1-1600*1200”). However, descriptions in a fixed value are recommended in order to reduce the processing load for an Imaging Initiator to determine the pixel size of an image.

Q2. Image-formats in capabilities object

A.

An Image-formats element is used by the Imaging Responder to inform an Imaging Initiator of the acceptable image format.

As the number of appearances of an image-formats element is specified as "*", in DTD of imaging-capabilities object, multiple descriptions of image format are possible. (It may be omitted when it does not act as an ImagePush Responder.) According to attribute-list declaration of image-formats element, three attribute values, “encoding”, “pixel”, and “maxsize” are defined. Among them, “encoding” is a mandatory attribute for the descriptions. “Pixel” may be omitted, though for the interoperability enhancement, the description is encouraged. When “pixel” is omitted, it shall be considered that there is no limitation in acceptable pixel sizes.

Shown below are an element type (ELEMENT) declaration and an attribute-list (ATTLIST) declaration related to an image-formats element extracted from DTD of an imaging-capabilities object.

<!ELEMENT image-formats EMPTY>
<!ATTLIST image-formats
encoding CDATA #REQUIRED
pixel CDATA #IMPLIED
maxsize CDATA #IMPLIED>

Description examples of an image-formats element are shown.

- Example1
  The Imaging Responder is able to receive JPEG encoded images of any pixel sizes.
  <imaging-capabilities version=”1.0”>
  <image-formats encoding=”JPEG” pixel=”1*1-65535*65535”/>
  </imaging-capabilities>

- Examble2
  The Imaging Responder is able to receive JPEG images of 1*1~1600*1200 pixel size.
  <imaging-capabilities version=”1.0”>
  <image-formats encoding=”JPEG” pixel=”1*1-1600*1200”/>
  </imaging-capabilities>

- Example3
  The Imaging Responder is able to receive only JPEG encoded images with sizes 160*120 pixels, 320*240, 640*480 and 1280*960.

<imaging-capabilities version=”1.0”>
<image-formats encoding=”JPEG” pixel=”160*120”/>
<image-formats encoding=”JPEG” pixel=”320*240”/>
<image-formats encoding=”JPEG” pixel=”640*480”/>
<image-formats encoding=”JPEG” pixel=”1280*960”/>
</imaging-capabilities>

- Example4
  The Imaging Responder is able to receive only JPEG encoded images with sizes 160*120~320*240 pixels, 640*480 and 1280*960.
  <imaging-capabilities version=”1.0”>
  <image-formats encoding=”JPEG” pixel=”160*120-320*240”/>
  <image-formats encoding=”JPEG” pixel=”640*480”/>
  <image-formats encoding=”JPEG” pixel=”1280*960”/>
  </imaging-capabilities>

As shown in above examples, when describing an image-formats element, acceptable pixel sizes of Imaging Responder may be described either by a range of pixel sizes (e.g. ”1*1-1600*1200”) or a fixed value (e.g. ”640*480”). However when a pixel size is described by a fixed value, an Imaging Initiator maybe able to resize the image to an approximate size but not to an exact size, and may fail to transmit it for a few pixels error. Therefore in order to enhance the interoperability, it is recommended to describe acceptable pixel sizes by a range of pixel sizes as long as Imaging Responder’s capability permits.

Q3. Example procedure to analyze capabilities object and its algorithm

A.

The graph below is an example of a procedure analyzing capabilities object in case of an Image Push initiator.

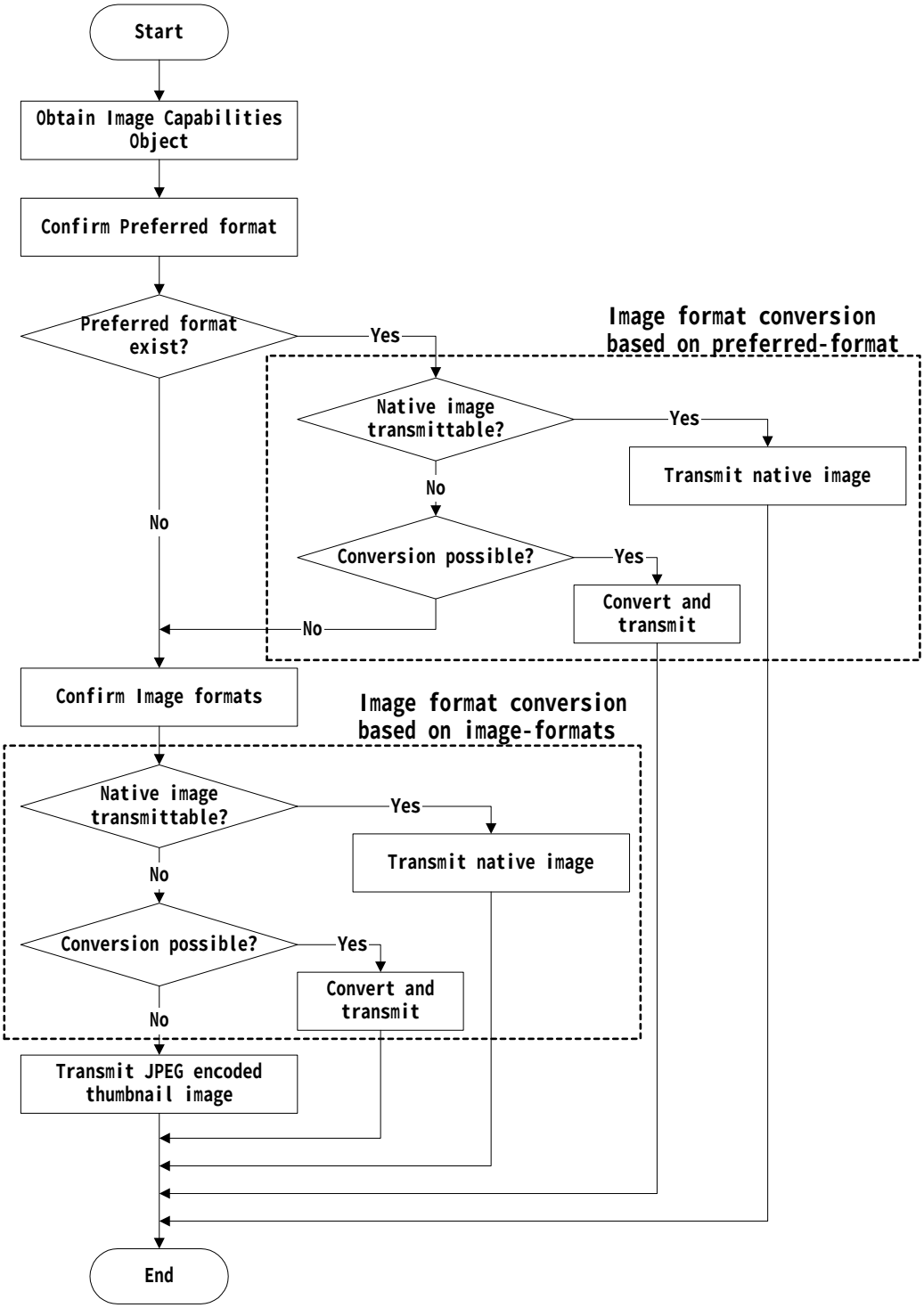


Figure 1: A recommended procedure to analyze Image Capabilities Object

Image format conversion based on a preferred-format:

According to the ATTLIST preferred-format, an original image should be transmitted without conversion if possible. If the result of the analysis reveals that the original image cannot be transmitted, the conversion is performed and the converted image is transmitted. When the conversion is impossible, the process of the image format conversion based on a preferred-format is terminated.

When the pixel size in preferred-format is described by a fixed value, it is recommended to resize the image to this fixed value if possible. When the pixel size in preferred-format is described using a range and the native image pixel size exceeds upper limit, it is recommended to resize the image to the upper limit. On the contrary when the native image pixel size falls below lower limit, it is recommended to resize the image to the lower limit. And when scaling down the image to a pixel size specified by an Imaging Responder while maintaining the aspect ratio, and if the size is not identical to the specified pixel size because of decimal point handling etc., it is recommended, if necessary, to apply a conversion processes such as fill, crop and stretch for the matching.

On the other hand, when a pixel size is not described in a preferred-format, it shall be decided referring to the value in the image-formats.

Image format conversion based on image-formats:

According to ATTLIST image-formats, an original image should be transmitted without conversion if possible. If native image format is not one of the formats listed in image-formats, conversion is performed and the converted image is transmitted. When the conversion is impossible, the process of image format conversion based on image-formats is terminated.

Resizing to a larger pixel sizes should be avoided as much as possible. It is recommended to resize to the biggest pixel size of Imaging Responder’s pixel sizes range, keeping it smaller than original image. And when reducing the image to the pixel size specified by an Imaging Responder while maintaining the aspect ratio, and if the size is not identical to the specified pixel size because of decimal point handling etc., it is recommended, if necessary, to apply a conversion processes such as fill, crop and stretch for the matching.

Q4. Descriptions of an image element in an Image Descriptor at the time of ImagePush

A.

When transmitting an image using ImagePush, it is mandatory to describe properties of the image to be transmitted in an image element of Image Descriptor. Image Descriptor is defined as follows:

```
<!DOCTYPE image-descriptor [
<!ELEMENT image-descriptor ( image ) >
<!ATTLIST image-descriptor version CDATA #FIXED ”1.0” >
<!ELEMENT image EMPTY>
<!ATTLIST image
    encoding CDATA #REQUIRED
    pixel CDATA #REQUIRED
    size CDATA #IMPLIED
    maxsize CDATA #IMPLIED
    transformation (stretch | fill | crop ) #IMPLIED
]>
```

According to the attribute-list declaration of the image element, five attribute values are defined; “encoding”, “pixel”, “size”, “maxsize” and “transformation”. Among them, “encoding” and “pixel” are mandatory attributes for the description. “Size” may be omitted, though for the enhanced interoperability, the use of this attribute is encouraged.

When an Imaging Initiator specifies the image size into the Image Descriptor, It enables the Imaging Responder to confirm the image size upon the image reception. Therefore, for example, when the Imaging Responder notices that the image size is larger than the reception buffer size, the Imaging Responder can immediately return an error code and terminate the processing.

Q5. Determent of data transmission instructed by an Imaging Responder to an Imaging Initiator at the time of Image Push

A.

In some cases when Image Push is used, an Imaging Responder may want to temporarily withhold data transmissions from an Imaging Initiator during a communication sequence; for example in a case that the Imaging Responder's storage area for received data becomes insufficient.

For example, in case of transmission of image file to a printer, the situation stated above happens when the Imaging Initiator continuously sends several pages or errors such as paper-out or ink-out occurs at a printer during the transmission. The communication sequence is as follows.

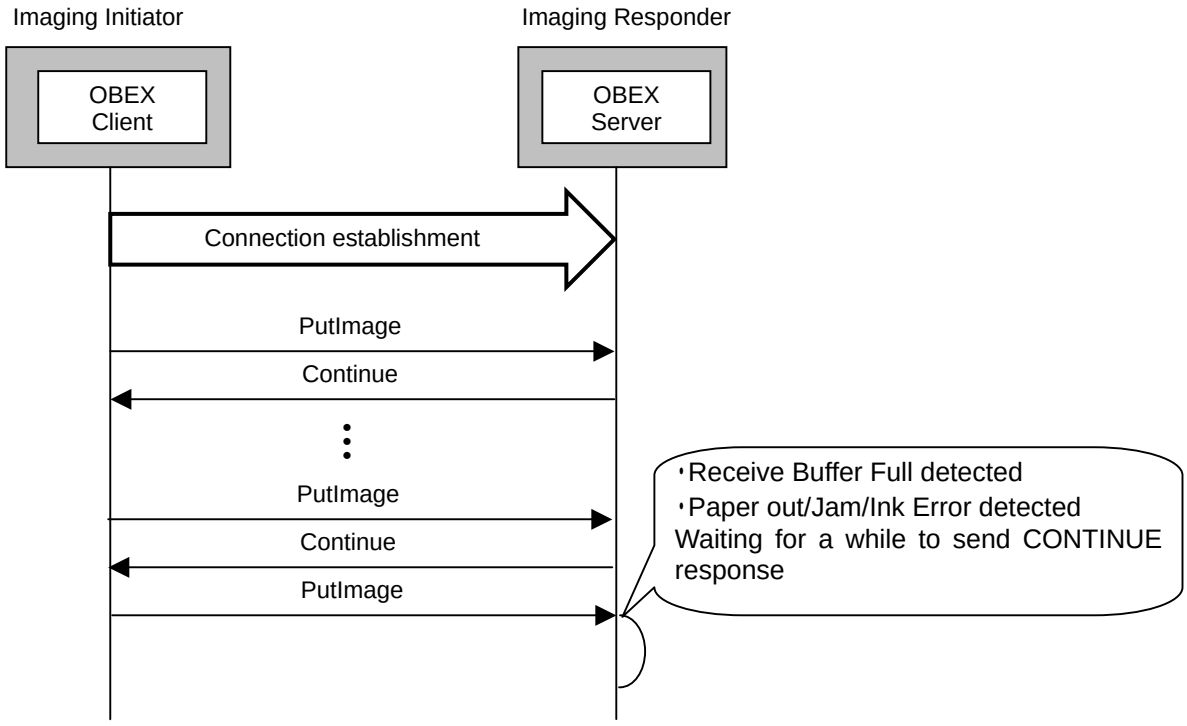


Figure 2: A communication sequence example where the Imaging Responder sustains response during ImagePush

Problems of the communication sequence in Figure 2 are listed below.

Problems on the Imaging Responder's side (e.g. printer) are;

- Printing may may not be performed due to disconnection if an Imaging Initiator has set a timer for the reception of the response.
- In case of an Imaging Responder implements to return an error code (ex. Not Acceptable/Service Unavailable), the Imaging Initiator could possibly release the session.

Impacts on the Imaging Initiator are;

- If an Imaging Initiator uses, for example, a progress bar indicating the progress of an image

transmission, the progress bar would stop due to lack of the response, and it would look like the application was frozen.

- When there is no response, an Imaging Initiator is unable to judge whether it is caused by flow control or some error in the Imaging Responder.

In such case, implementations stated below are recommended.

- An Imaging Responder suspends the response of OBEX PUT command.
- An Imaging Initiator sets longer timeout value
- An Imaging Initiator release the session after timeout.

Figure 3 shows the communication sequence in case of timeout of the Imaging Initiator.

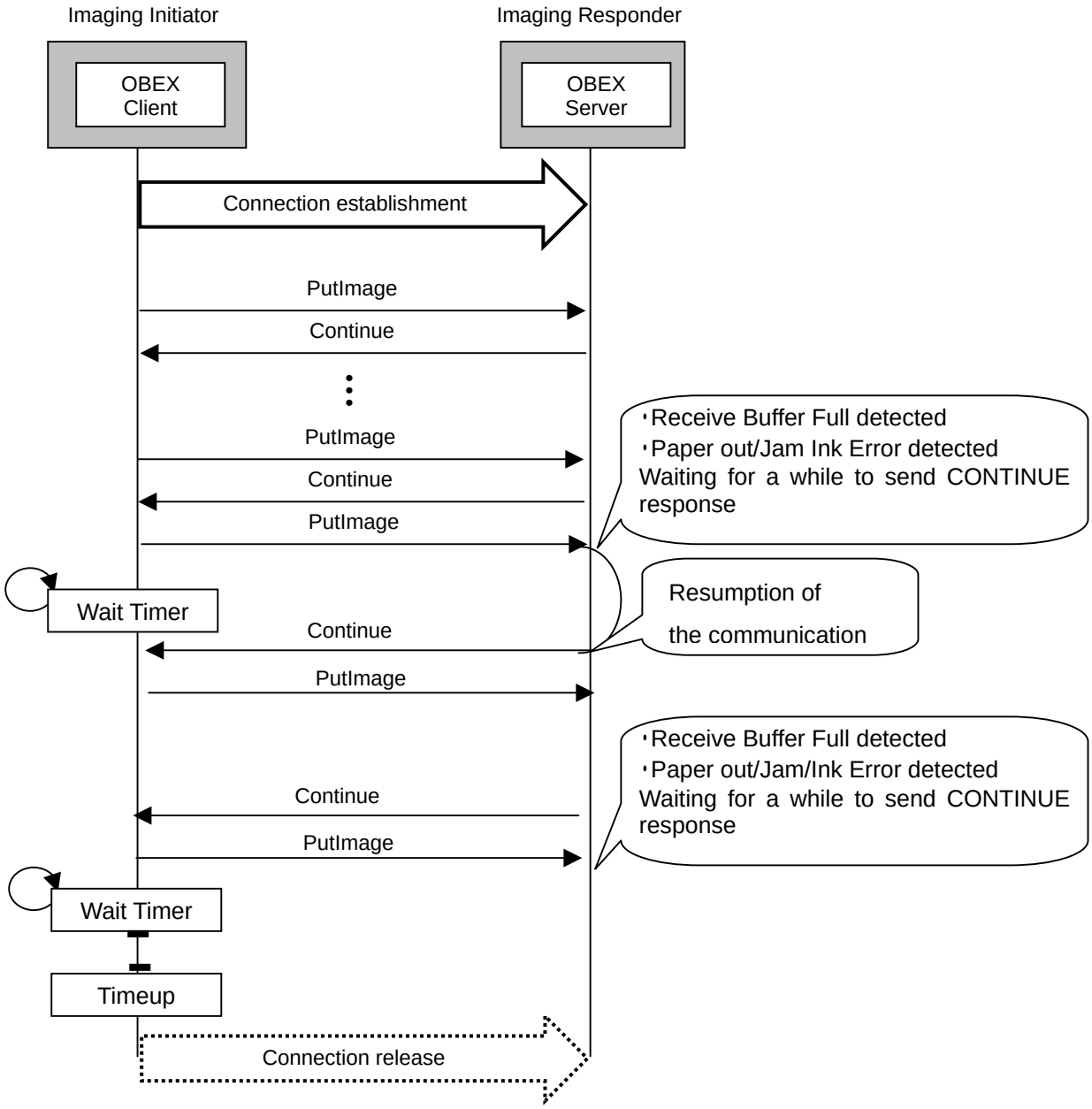


Figure 3: A Communication Sequence Example for the Recommended Implementations in Image Push

Timeout is Imaging Initiator's implementation dependent. Examples of timeout setting of an Imaging Initiator when transmitting image files to a printer are shown below as references:

Example 1: Considering the possibility of paper out or ink out of the printer.
User shall be given enough time to restore the printer before timeout, although it depends on an actual usage situation.

Example 2: Considering possibility of consecutive jobs using same session to an inkjet printer.
Timeout value shall be set in consideration of time necessary for an inkjet printer to print one page of image. In case of an Imaging Initiator is likely to perform consecutive printings, timeout value shall be set in consideration of a printing time. (Setting timeout value shorter than the duration of one page print would possibly generate an error during consecutive printings)

Example 3: In case timeout value is set giving highest priority on power saving of an Imaging Initiator.
Timeout value shall be set in consideration of the battery capacity of an Imaging Initiator.

* When communications are made between the Imaging Initiator and the Imaging Responder as described in Q4 or Q12, the possibility to cause such problem is low.

Q6. Response code of GetStatus Request

A.

Paragraph 4.5.15 of BIP[1] describes response codes of GetStatus Request as follows:

- After establishment of the secondary connection, an Imaging Responder shall return a response code “Continue” to each GetStatus request coming from an Imaging Initiator. After the secondary connection has been disconnected, an Imaging Responder shall return “Success” response code.
- In case of secondary connection cannot be preserved due to an error on the Imaging Responder, an error response code should be returned to GetStatus request coming from the Imaging Initiator.

Furthermore, it is recommended for an Imaging Responder to implement the following in order to enhance the interoperability.

- An Imaging Responder shall return “Continue” response code to a GetStatus request coming from the Imaging Initiator before the establishment of the secondary connection.

Figure 4 shows a communication sequence using the additional recommendation.

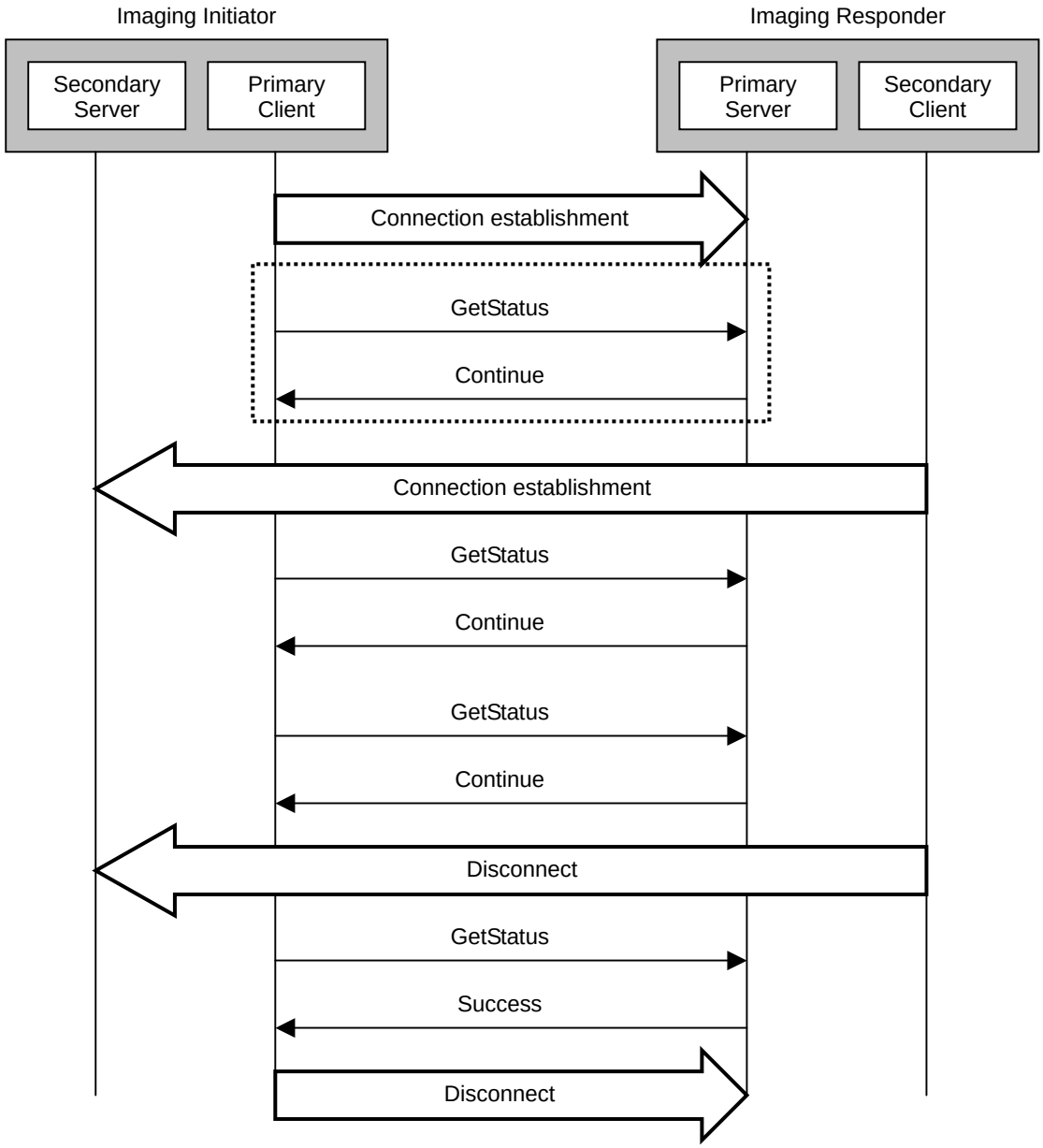


Figure 4: A communication sequence example using recommendation with Advanced Image Printing

The explanation is as follows:

Regarding the portion inside dotted line in Figure 4, the Imaging Responder shall assume that a Primary Client of the Imaging Initiator may or may not transmit GetStatus request. (BIP[1] specifies transmission condition for GetStatus request in StartArchive but does not specify it in StartPrint.) For example, if an Imaging Responder is implemented to return an error code or “Success” response code when the secondary connection is not established, the Imaging Initiator would release the primary connection just before or during the establishment process of the secondary connection. (BIP[1] specifies that upon receiving “Success”, the Imaging Initiator shall release the primary connection.) Figure 5 shows the communication sequence described above.

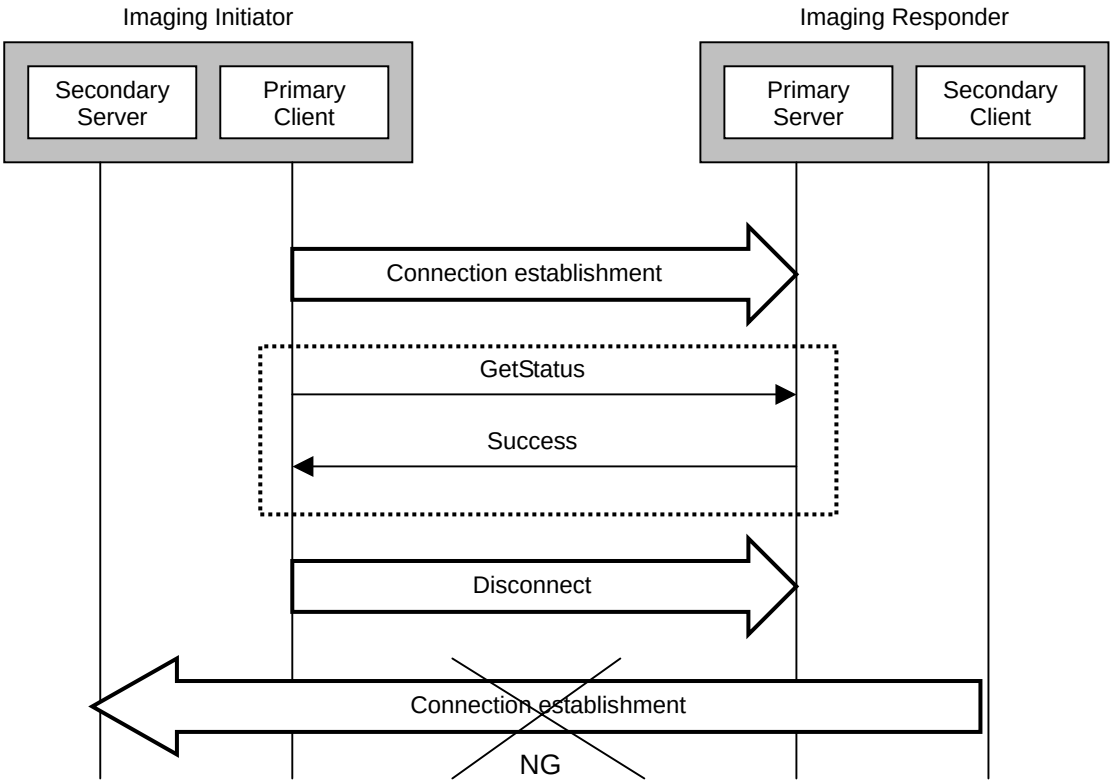


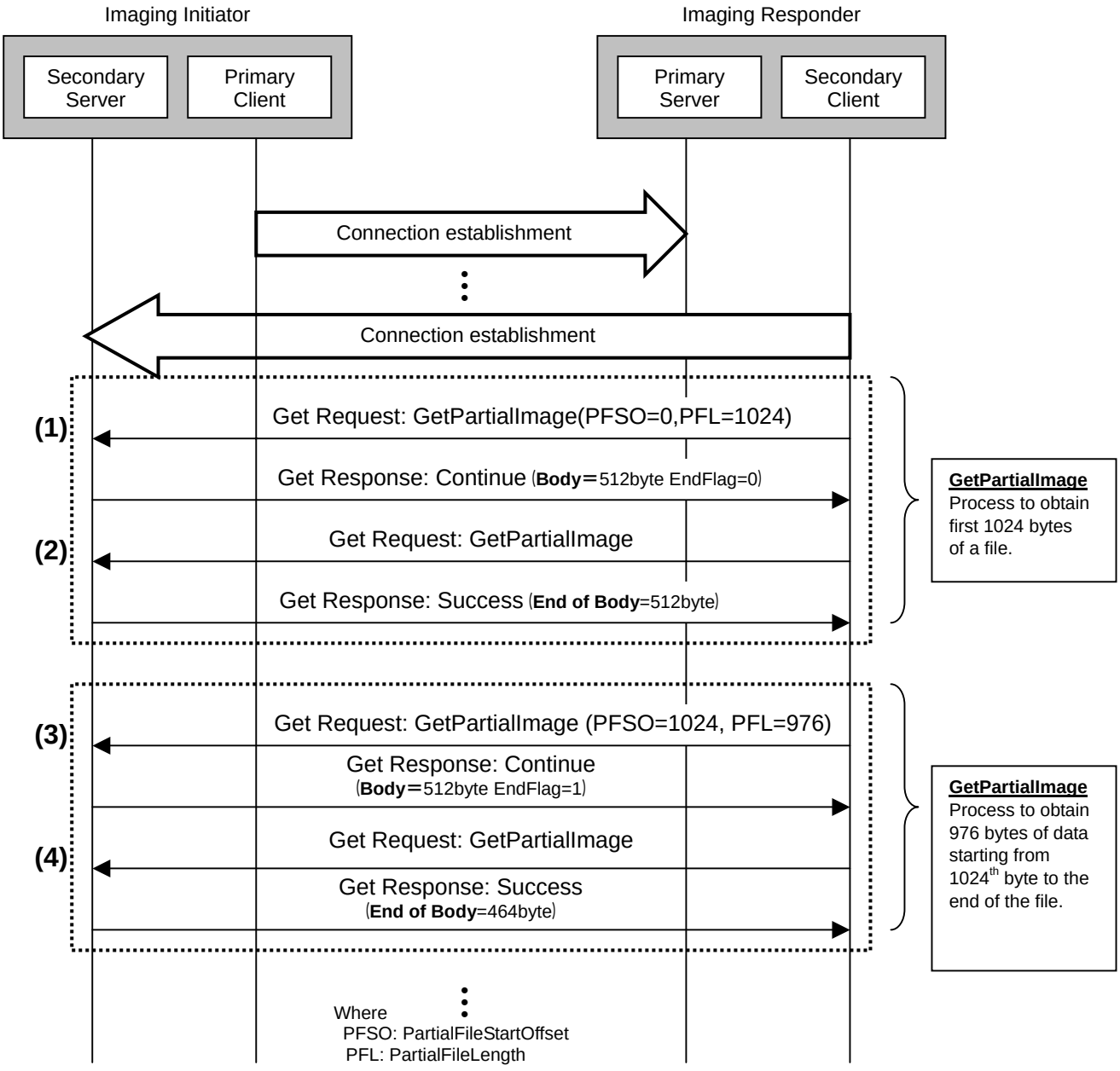
Figure 5: Possible problems in case the recommended implementation is not performed

Q7. How to use Body/End of Body headers with GetPartialImage

A.

When transmitting a partial image file requested by GetPartialImage, an Imaging Initiator is recommended to use an End of Body header in case of the final part of the partial image file, but use Body header in the other cases. And in case of the response of the final part of the image file, EndFlag=1 should be set.

Shown below is an example of Body / End of Body headers usage when obtaining a 2000 bytes JPEG image (img5.jpg) using GetPartialImage.



Secondary Client Request:	**Bytes**	**Meaning**
(1)Opcode	0x83	**GET**, Final bit set
	0xnnnn	Length of packet
	0xCB	HI for **Connection Id** header
	0xnnnnnnnn	ConnId = nnnnnnnn
	0x42	HI for **Type** header
	0x0014	Length of **Type** header
	x-bt/img-partial	MIME Media-Type of object, null terminated
	0x01	HI for **Name** header
	0x0015	Length of **Name** header
	0069 006D 0067	**Name** header content “img5.jpg”
	0035 002E 006A	(UTF-16 encoded), null terminated
	0070 0067 0000	
	0x4C	HI for **Application Parameter** header
	0x000F	Length of **Application Parameter** header
	04 04 00000400	PartialFileLength (= 1024 Bytes)
	05 04 00000000	PartialFileStartOffset (= 0)
Secondary Server response:		
(1)Response code	0x90	**Continue,** Final bit set
	0xnnnn	Length of response packet
	0xC3	HI for **Length** header
	0x00000400	Length of partial file
	0x4C	HI for **Application Parameter** header
	0x000C	Length of **Application Parameter** header
	06 04 000007D0	TotalFileSize (= 2000 Bytes)
	07 01 00	EndFlag (= FALSE)
	0x48	HI for **Body** header
	0x0200	Length of **Body** header
	0x……	
Secondary Client Request:		
(2)Opcode	0x83	**GET**, Final bit set
	0xnnnn	Length of packet
	0xCB	HI for **Connection Id** header
	0xnnnnnnnn	ConnId = nnnnnnnn
Secondary		

Server response:		
(2)Response code	0xA0 0xnnnn 0x49 0x0200 0x……	**Success,** Final bit set Length of response packet HI for **End of Body** header Length of **End of Body** header

Secondary Client Request:	**Bytes**	**Meaning**
(3)Opcode	0x83 0xnnnn 0xCB 0xnnnnnnnn 0x42 0x0014 x-bt/img-partial 0x01 0x0015 0069 006D 0067 0035 002E 006A 0070 0067 0000 0x4C 0x000F 04 04 000003D0 05 04 00000400	**GET**, Final bit set Length of packet HI for **Connection Id** header Connld = nnnnnnnn HI for **Type** header Length of **Type** header MIME Media-Type of object, null terminated HI for **Name** header Length of **Name** header **Name** header content “img5.jpg” (UTF-16 encoded), null terminated HI for **Application Parameter** header Length of **Application Parameter** header PartialFileLength (= 976 Bytes) PartialFileStartOffset (= 1024)
Secondary Server response:		
(3)Response code	0x90 0xnnnn 0xC3 0x000003D0 0x4C 0x000C 06 04 000007D0 07 01 01 0x48 0x0200 0x……	**Continue,** Final bit set Length of response packet HI for **Length** header Length of partial file HI for **Application Parameter** header Length of **Application Parameter** header TotalFileSize (= 2000 Bytes) EndFlag (= TRUE) HI for **Body** header Length of **Body** header

Secondary Client Request:		
(4)Opcode	0x83 0xnnnn 0xCB 0xnnnnnnnn	**GET**, Final bit set Length of packet HI for **Connection Id** header Connld = nnnnnnnn
Secondary Server response:		
(4)Response code	0xA0 0xnnnn 0x49 0x01D0 0x……	**Success,** Final bit set Length of response packet HI for **End of Body** header Length of **End of Body** header

Figure 6: An Example of Body/End of Body headers usage in Advanced Image Print

Q8. Image formats transmitted to a digital still camera etc.

A.

As there are various types of JPEG formats, digital still cameras may not handle all of those formats properly. Even if transmissions of a JPEG encoded image succeed, an Imaging Responder might not be able to display this image. Recommendations for transmissions of JPEG encoded image to a digital still camera etc. are as follows:

Digital still camera etc. generally use image formats which conforms to Exif[3] and DCF[4]. When other JPEG encoded images are sent to a digital still camera etc., the main images or the thumbnail images may not be displayed properly. Therefore, it is recommended to do as follows:

1) If necessary, before transmitting JPEG images, an Imaging Initiator should conform them to Exif[3] and DCF[4].

(A judgment of whether Imaging Responder is digital still camera or not can be made by the CoD.)

2) In other cases, it is recommended to transform an image that satisfies following conditions.

- Use standard Huffman table [5].
- Use baseline DCT method [5].

Furthermore, when a digital still camera acts as an Imaging Responder, the possibility to receive non-Exif[3] and DCF[4] compliant images should be taken into considerations.

Q9. When a file name cannot be handled by Imaging Responder file system

A.

In some cases, Name header cannot be handled by a file system of an Imaging Responder. For example, an Imaging Initiator may transmit an image with a Name header encoded by Japanese character set, while an Imaging Responder cannot handle Japanese characters.

This type of situation is also described in Q14 of BIP[1] whitepaper[2]. For higher interoperability, It is recommended to proceed as follows:

- When the file name of received image cannot be handled by the file system, the Imaging Responder should proceed to create file after converting the file name to a usable one.

Q10. Maximum OBEX packet size

A.

In OBEX[6], maximum OBEX packet sizes are negotiated during the connection process, and the smaller one is adopted as their communication maximum packet size for their onward communications. (The usable packet size based on OBEX[6] specification is from 255bytes to 64Kbytes-1.)

When this size is small, transmission rate becomes extremely slow. For example, in case of PUT operation, the OBEX Server returns a response, following the OBEX Client data. The OBEX client does not transmit next data until it receives a response for the previously transmitted data. The time waiting for responses creates a big overhead. When the packet size is large, the frequency of a response wait becomes smaller and throughput is improved.

Because BIP[1] handles images, the amount of transmitted data is large. The packet size should be as large as possible in order to have good throughput. Recommended packet size is the one that enables transmission of a whole thumbnail image at one time. (e.g. 5Kbytes)

Q11. Supported features attributes of service record when supporting Image

A.

Refer to 6.1.1. of BIP[1].

Q12. Usage of Total image data capacity of service record

A.

Total image data capacity attribute is a reference value of the memory size of the Imaging Responder for the image storage capability. For example, although the memory size for image storage capability decreases as images are received, it is not necessary to dynamically change the value of the Total image data capacity.

An Imaging Initiator can decide not to transmit an image in case that the size of this image is bigger than the Total image data capacity of the Imaging Responder.

However, since the value of a Total image data capacity is just a reference, there is no guarantee that the Imaging Responder will succeed in storing an image transferred from an Imaging Initiator, even if the Total image data capacity is larger than the size of the image.

Q13. Usage of the Class of Device/Service (CoD)

A.

CoD is a 3 octets value and can be described as

b_{23} b_{22} b_{21} b_{20} b_{19} b_{18} b_{17} b_{16} b_{15} b_{14} b_{13} b_{12} b_{11} b_{10} b_9 b_8 b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0.

Where "b_{23} b_{22} b_{21} b_{20} b_{19} b_{18} b_{17} b_{16} b_{15} b_{14} b_{13}" represents the Major Service Class, "b_{12} b_{11} b_{10} b_9 b_8" represents the Major Device Class and "b_7 b_6 b_5 b_4 b_3 b_2" represents the Minor Device Class. "b_1 b_0" is the Format Type field and should have the value "0 0". Each bit should be set in conformance with Assigned Numbers Specification[7].

The value of the Major Service Class is subject to change depending on available functions (services), but it is recommended that the Object Transfer attribute bit "b_{20}" is set to 1. CoD examples with the descriptions are shown below.

Device type

Major Service Class: b_{23} b_{22} b_{21} b_{20} b_{19} b_{18} b_{17} b_{16} b_{15} b_{14} b_{13}

Major Device Class: b_{12} b_{11} b_{10} b_9 b_8

Minor Device Class: b_7 b_6 b_5 b_4 b_3 b_2

Digital Still Camera

Major Service Class: 0 0 0 **1** 1 0 0 0 0 0 0

Major Device Class: 0 0 1 1 0

Minor Device Class: 0 0 1 0 0 0

Printer

Major Service Class: 0 0 0 **1** 0 1 0 0 0 0 0

Major Device Class: 0 0 1 1 0

Minor Device Class: 1 0 0 0 0 0

Cellular Phone

Major Service Class: 0 1 0 **1** 0 0 0 0 0 0 0

Major Device Class: 0 0 0 1 0

Minor Device Class: 0 0 0 0 0 1

Laptop PC

Major Service Class: 0 0 0 **1** 0 0 0 0 0 0 0

Major Device Class: 0 0 0 0 1

Minor Device Class: 0 0 0 0 1 1

Handheld PC/PDA

Major Service Class: 0 0 0 **1** 0 0 0 0 0 0 0

Major Device Class: 0 0 0 0 1

Minor Device Class: 0 0 0 1 0 0

Palm sized PC

Major Service Class: 0 0 0 **1** 0 0 0 0 0 0 0

Major Device Class: 0 0 0 0 1

Minor Device Class: 0 0 0 1 0 1

Contributors

Kazuo Shimizu		OLYMPUS CORPORATION
Yasuji Tamatsu		OPEN INTERFACE, INC.
Hideaki Tanaka		OPEN INTERFACE, INC.
Yasumasa Takahashi		OPEN INTERFACE, INC.
Soichi Saito		DENSO CORPORATION
Kiyotaka Inaba		RICOH COMPANY, LTD.
Hiroaki Uchiyama		RICOH COMPANY, LTD.
Alain Volmat		RICOH COMPANY, LTD.
Takuya Kawamura	Leader	TOSHIBA CORPORATION
Takayasu Aoki		TOSHIBA CORPORATION Digital Media Network Company
Kenichi Fujii	Sub-leader	CANON INC.
Kenichi Kurata		SEIKO EPSON CORPORATION
Tomohiro Ogawa		Sony Corporation
Mitsuyoshi Yasuda		Sony Ericsson Mobile Communications Japan, Inc.
Hiroshi Kato		TAIYO YUDEN CO., LTD.
Katsumi Murakami		FUJITSU LIMITED/FUJITSU DEVICES INC.
Yoshio Watanabe		FUJITSU LIMITED/FUJITSU DEVICES INC.
Tatsuya Inagaki		Matsushita Electric Industrial Co., Ltd.