ENGLISH TRANSLATION

DSRC BASIC APPLICATION INTERFACES SPECIFICATION GUIDELINE

ITS FORUM RC-004 Version 1.2

Version 1.1 March 28, 2007
Version 1.2 October 11, 2011

ITS Info-communications Forum of Japan

General Notes to the English Translation of ITS Info-communications Forum Guidelines

1. Notes on Copyright

- The copyright of this document is ascribed to the ITS Info-communications Forum of Japan.

- All rights reserved. No part of this document may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, without the prior written permission of ITS Info-communications Forum of Japan.

2. Notes on English Translation

- ITS Info-communications Forum Guidelines are usually written in Japanese. This document is a translation into English of the original document for the purpose of convenience of users. If there are any discrepancies in the content, expressions, etc. between the original document and this translated document, the original document shall prevail.

- ITS Info-communications Forum Guidelines, in the original language, are made publicly available through web posting. The original document of this translation may have been further revised and therefore users are encouraged to check the latest version at an appropriate page under the following URL:
http://www.itsforum.gr.jp/Public/Eguideline/index.html.

DSRC BASIC APPLICATION INTERFACES SPECIFICATION GUIDELINE

ITS FORUM RC-004 Version 1.2

Version 1.1 March 28, 2007
Version 1.2 October 11, 2011

ITS Info-communications Forum of Japan

Revision History

Version	Date	Status	Summary of changes
1.1	Mar. 28, 2007	Establishment	English Translation
1.2	Oct. 11, 2011	Function addition & Error in writing correction	Contactless IC smart card , Push-type information delivery application(Payment Application type, dsrc privateSpot image's Contents Type) addition

FORWARD

This guideline aims at the following to promote further diffusion of the DSRC.

(1) Improving serviceableness for users of DSRC On-Board Equipment (OBE)

- Install the basic application interface to make it possible to provide low-price OBE with a fewer resources.

- Install the basic application interface to greatly increase opportunities for the users to operate their OBE and to improve serviceableness while promoting adaptation of intelligent features.

(2) Improving ease of building up infrastructure

- Reduce expenses and time for building up the infrastructure without manufacturing or selling a dedicated DSRC OBE compatible with the application supplied.

- Provide an application made up of the basic application interfaces to make the DSRC OBE on the market the potential targets of the services, thus enabling to provide more fine-tuned services.

(3) Examples of applying basic application interfaces for services

- "Examples of applying basic application interfaces for services" is provided as a supplement document. It is desired that the system design for building up infrastructure incorporates the documented data.

- It is also desired that, where conducting the DSRC road-to vehicle mutual connectability tests, the communications transactions shown in "Examples of applying basic application interfaces for services" be utilized as an example of the application test transaction.

DSRC BASIC APPLICATION INTERFACES SPECIFICATION GUIDELINE

Table of Contents

Chapter 1 General....1
1.1 Overview....1
1.2 Scope Status of Application of This Document....1
1.3 Reserve Field....3
1.4 Coding Rule....3
1.5 References....3
Chapter 2 Basic Primitive Application Function Overview....5
2.1 Assumption of Basic Primitive Application Functions....5
2.2 Classification of Basic Primitive Application Functions....6
2.3 Functional Overview of Basic Application Interface....7
2.4 Local Port Number of Basic Application Interface....10
Chapter 3 Basic Application Interfaces Specification....13
3.1 OBE Instruction Response Application....13
3.2 OBE Memory Access Application....27
3.3 IC Card Access Application....100
3.4 Push-type Information Delivery Application....118
3.5 OBE ID Communication Application....161
3.6 OBE Basic Indication Application....192
Annex A Relationship between DSRC Services and Basic Application Interface....201
A.1 Postulated Examples of DSRC Services and Required Functions....201
A.2 Detailed Function Analysis....202
Annex B Relationship with Security Platform....212
B.1 Security Platform Configuration....212
B.2 Local Port Number List....213
B.3 Important Notice about Basic Application Interface using SPF....215
Annex C Service Application Examples using Basic Application Interface....216
C.1 Settlement Processing using OBE ID....216
C.2 Prepaid Type Settlement Processing using IC Card Access....218
C.3 Settlement Processing using IC Card ID....221
C.4 Information Providing Service using Push-type Information Delivery....224

C.5 Settlement Processing using IC Card 228
Annex D OBE Memory Access Application 230
D.1 Memory Tag Configuration 230
D.2 Important Notice about Options 231
D.3 Protection Mode 235
D.4 Memory Access Management 235
Annex E Implementation Example and Important Notice about OBE ID Communication Application 241
E.1 Example when DSRC-SPF is adopted as In-application Security 241
E.2 Important Notice when In-application Security is not handled 247
E.3 Important Notice for handling In-application Security 247
Annex F Version of Basic Application Interface 249
F.1 Definition 249
F.2 Intended Purpose 249
F.3 Elements of Version Management 249
F.4 Version Update 249
F.5 Compatibility 250
F.6 Selection of Version 250
Annex G Push-type Information Delivery Application 252
G.1 Client Information Notification Command [Informative] 252
G.2 Important Notice about Push-type Information Delivery Application 252
G.3 Operation Example of OBE triggered by RC/DC Flag 253
Amendment History 255
Amendment item1 Amendment List about problem of PushID 255
Amendment item2 Amendment List about Addition of Application Type / Content Type 258
Amendment item3 Amendment List about Client Information Notice Command 260
Amendment item4 Amendment List about Others 261
Amendment History 263
Amendment item1 List about Addition of Reference 263
Amendment item2 List about Correction of Misdescription 264
Amendment item3 Amendment List about Addition of Contactless IC Smart Card-compliance 265
Amendment item4 Amendment List about Addition of Application Type and Content Type .266

Chapter 1 General

1.1 Overview

The Dedicated Short-Range Communication (DSRC) system, which is characterized as a "short-range, small-zone bi-directional mobile communication compatible with multiple applications", is one of the core systems in the intelligent transport systems or ITS. Some of the diversified, variegated services utilizing the DSRC technology have already been in commercial operation in the Electronic Toll Collection system (ETC). As applied services other than for the ETC system, the parking lot DSRC system, gas station DSRC system, drive-through DSRC system and the DSRC information services to running vehicles are becoming available.

As described above, the increasing demand for and compatibility with the provision of non-IP, multi-functional services using the DSRC system today greatly expands the opportunities to utilize on-board DSRCs. This in turn requires the on-board DSRC to be compatible with multiple functions. However, since the on-board DSRCs in the early stages of diffusion have a few resources, it is difficult to deal with the applications that may possibly be used in future.

This guideline first defines the basic, necessary applications, then, specifies the interface portions of these basic applications between the group of diversified applications and the non-IP local port protocol of the application sub-layers (ARIB STD-T88) specified by the Association of Radio Industries and Businesses (ARIB). With the concept that some of these are combined to be compatible with a variety of services, the basic application interface specifications are defined.

Based on this concept, this guideline therefore defines the system necessary to realize a higher level services while supporting the human machine interfaces (HMI) of the OBE that can only have a minimum resources and covering the applications unique to the DSRC.

1.2 Scope Status of Application of This Document

1.2.1 Scope

DSRC systems that this guideline applies consists of the DSRC road side system, DSRC OBE and test unit defined in the standard specifications ARIB STD-T75 and ARIB STD-T88.

Within this DSRC system, this guideline defines the road-to-vehicle communications interface of the following DSRC basic application interfaces.

(1) OBE instruction response application
(2) OBE memory access application
(3) IC card access application
(4) Push-type information delivery application
(5) OBE ID communication application
(6) OBE basic indication application

1.2.2 Position of basic application interfaces in protocol configuration

Figure 1.2-1 shows the position of the basic application interfaces in the protocol configuration.

To prevent information leakage during exchanging personal information and fare settlement data and to fend off attacks to the system by an ill-willed road side system or OBE, mutual identification and equipment authentication should take place. To this end, the security platform needs to build on the local port protocol of the application sub-layer (ASL). The relationships between this security platform and the basic application interfaces are also described (Annex. B).

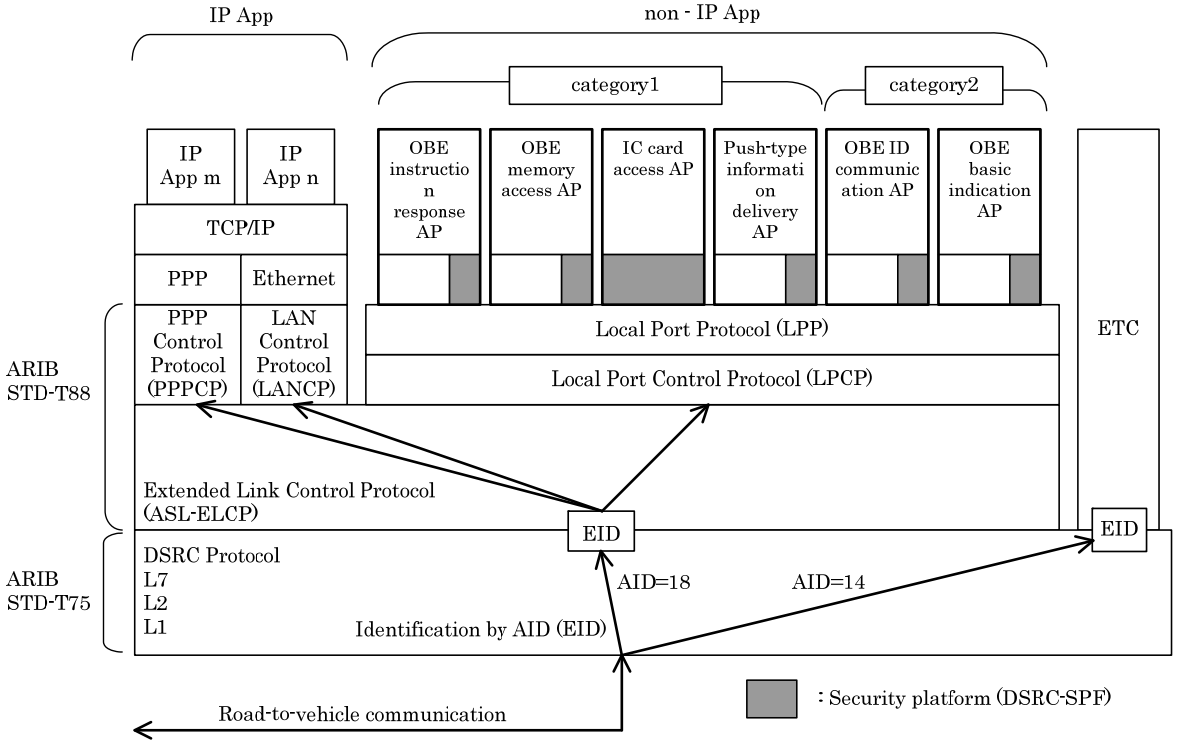


Figure 1.2-1 Basic application interface protocol configuration

(Note) Refer to clause 2.2 for categories

1.3 Reserve Field

The variables or information filed as defined “reserve” in this guideline are reserved for future expansion capabilities. These variables or information filed as defined “reserve” in this guideline (version) may be defined as specific values or identifiers. The user of this guideline should take into account that these values or identifiers might be changed in the future version.

1.4 Coding Rule

Variables specified in this guideline are described using Abstract Syntax Notation One :ISO/IEC 8824). The coding rules are a packed encoding rule (UNALIGNED PER (Packed Encoding Rule: ISO/IEC 8825-2).

1.5 References

This Guideline incorporates provisions from other publications by dated or undated reference. These normative references are cited at the appropriate places in the text and the publications are listed hereafter. For dated references, subsequent amendments to or revisions of any of these publications apply to this Guideline only when incorporated in it by amendment or revision. For undated references the latest edition of the publication referred to applies.

ARIB STD-T75	DEDICATED SHORT-RANGE COMMUNICATION (DSRC) SYSTEM
ARIB TR-T16	DEDICATED SHORT-RANGE COMMUNICATION (DSRC) SYSTEM TEST ITEMS AND CONDITIONS FOR LAND MOBILE STATION COMPATIBILITY CONFIRMATION
ARIB STD-T88	DEDICATED SHORT-RANGE COMMUNICATION (DSRC) APPLICATION SUB LAYER
ARIB TR-T17	TEST ITEMS AND CONDITIONS FOR DEDICATED SHORT-RANGE COMMUNICATION (DSRC) APPLICATION SUB LAYER LAND MOBILE STATION COMPATIBILITY CONFIRMATION
ISO/IEC 8824-1	Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation
ISO/IEC 8825-2	Information technology - ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)
JEITA TT-6001A	Standard specification of ITS On-Board Unit
JEITA TT-6002A	Standard specification for DSRC section of ITS On-Board Unit
JEITA TT-6003A	Standard specification for Car Navigation System section of ITS On-Board Unit
JEITA TT-6004	Speech synthesizer symbols for ITS on-Board Unit
ISO15628:2007	Road transport and traffic telematics -- Dedicated short range communication (DSRC) -- DSRC application layer
ISO24103:2009	Intelligent transport systems -- Communications access for land mobiles (CALM) -- Media adapted interface layer (MAIL)
ISO29281:2011	Intelligent transport systems -- Communications access for land mobiles (CALM) -- Non-IP networking

Chapter 2 Basic Primitive Application Function Overview

2.1 Assumption of Basic Primitive Application Functions

The Basic primitive application functions are required to correspond to the diversified services.

This section shows the basically-required functions and the specifications based on the philosophy, combining the several applications to correspond to the diversified services. Under this concept, what types of basic primitive application functions are defined as "basic" is important. This section specifies the structure which is required to realize the higher service, as same as to support the minimum amount of the human machine Interfaces (HMI) and to include the DSRC-specific application.

(1) Human Machine Interface functions of OBE

Because of OBE's resources, HMI functions loaded on OBE is limited.

Table 2.1.1 shows examples of the configuration of HMI of OBE in view of numbers and characters, voice, and sound) and display system (buttons).

Table 2.1.1 Examples of OBE configuration

	Example1	Example2	Example3	Example4
No HMI	X			
Display		X		X
Button			X	X
Required functions	none	Instruction	Indication	Instruction and response

Example 1 is the simple OBE without an HMI.

Example 2 uses resources known from existing ETC OBEs. It can provide number/character information, simple sound data, and alarm messages to the users. The application to realize these functions is defined as"OBE basic instruction application".

Examples 3 and 4 enable to transmit a message from mobile users to the road. This allows the users to transmit their decision such as "OK". The application to realize this function is defined as "OBE instruction response application".

More complicated services are realized by higher DSRC OBE with using the road-to-vehicle communication function described in section (3) below.

(2) DSRC- specific application

This section specifies the DSRC-specific application which takes advantage of the features of DSRC.

DSRC has various services which identifies OBE on the road side. By defining the OBE IDs available for many services, identification of OBE is realized. The application to realize this function is defined as “OBE ID communication application.”

For the application of DSRC, billing and settlement are assumed to be the main application next, and the IC card access is realized for the basic operation. The application to realize this operation is defined as “IC card access application.”

(3) Improving of convenience of road-to-vehicle communication function

The basic of DSRC is road-to-vehicle communication. Directly using LPCP and LPP realizes the road-to-vehicle communication. However, improving usability and convenience will make configuring the various services easier and more effective.

OBE Memory Access Application” is the basic function, defined aiming to easily create the flow of the information from RSU to OBE memory.

For the transmission of data from road to vehicles, it is assumed that the type of data will be diversified (multimedia such as sound and image as same as text). The "push-type information delivery application" which packages diversified data to transmit from road to vehicles is, therefore, defined.

2.2 Classification of Basic Primitive Application Functions

Six basic primitive applications specified by the DSRC basic primitive application interface specifications are classified by two categories, Category 1 and 2.

In Category 1, the applications which provide the access control function to correspond to the basic DSRC service scene are belonged. In Category 2, the applications which provide the access control function dedicated to individual organization and the application whose functions are included in the application in Category 1 are belonged. The applications of each category are listed below:

(1) Category 1

(a) OBE instruction response application

(b) OBE memory access application

(c) IC card access application

(d) Push-type information delivery application

(2) Category 2

(a) OBE ID communication application

(b) OBE basic indication application

2.3 Functional Overview of Basic Application Interface

This section describes the outline of functions of the six basic applications for the basic application interface defined in Section 2.1.

Figure 2.3.1 shows the functional configuration of the application to be realized by the local port protocol.

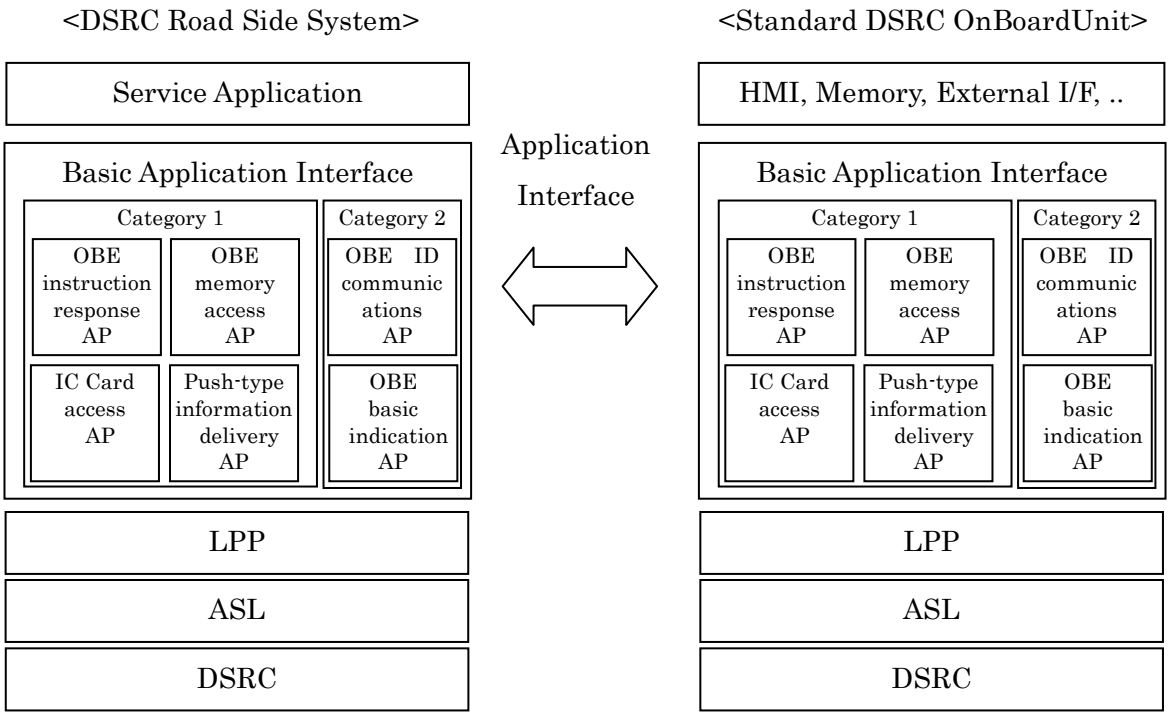


Figure 2.3.1 Functional configuration

2.3.1 OBE instruction response application

OBE instruction response application notifies the OBE of the specific instruction information from the external server connected to the road side equipment and returns the response from

the users by using the input mechanism (such as buttons) of OBE.

2.3.2 OBE memory access application

OBE memory access application is the application of the system on the roadside. The memory on OBE of the application stores variable length data in free form with search tag. This application also enables to write data to the mentioned memory of OBE and read from the application of the system on the roadside. In this case, the search tag needs to be preliminarily specified by application.

2.3.3 IC card access application

IC card access application provides the function to access the IC card on request from the system on the roadside by the method prescribed by ISO/IEC7816. The function of the IC card access application is exclusively applicable to the ISO/IEC7816-compliant IC card and contactless IC smart card.

2.3.4 Push-type information delivery application

Push-type information delivery application sends the contents or the position of the contents to the client on the OBE from the server of the system on the roadside. This application on the client side automatically executes the processing corresponding to each receive contents. The method that the contents are distributed is called "contents push," and the method that the position of contents (URL, etc.) is distributed, and the acquisition of the contents is separately executed with HTTP, etc. is called "pseudopush."

2.3.5 OBE ID communications application

OBE ID communications application notifies the roadside of the ID of the OBE to identify the OBE on the roadside. To communicate the OBE ID with road-to-vehicle communication, the system on the roadside notifies the OBE of the acquirer ID, and the OBE returns the OBE ID corresponding to the acquirer ID.

2.3.6 OBE basic indication application

OBE basic indication application is used to provide the minimum HMI function. The specific instruction data is notified to OBE from the external server connected to the roadside equipment.

2.3.7 Examples of basic primitive application functions

Table 2.3-1 shows the example of the combination of the basic primitive application functions and correspondence to DSRC services.

Table 2.3-1 Functions required for the DSRC services and correspondence to the basic primitive application functions

			OBE instruction response	OBE memory access	IC card access	Push-type information dielivery	OBE ID communication	OBE basic instruction
Fare settlement	Link settlement system	Function to give the instructions to OBE (as well as ETC system or expanded function)	X					X
		Function to access the OBE-specific information (ID)		X			X	
		Function to access the memory of OBE (accumulation of information for use)		X				
	Card transaction system	Function to give the instructions to OBE (as well as ETC system or expanded function)	X					X
		Function to confirm response of user	X					
		Function to access the IC card loaded into OBE			X			
		Function to access the memory of OBE (accumulation of information for use)		X				
Transmitting/receiving data on service system with IP	Vehicle-specific information controlsystem	Function to give the instructions to OBE (as well as ETC system or expanded function)	X					X
		Function to access the OBE-specific information (ID)		X			X	
		Function to access the memory of OBE (accumulation of information for use)		X				
		Function to access the information in server by request from OBE on IP (Providing request/response type information)						

		Function to distribute the URL of the start page by the information providing service and other information from the information providing server to OBE (providing of push-type information)				X		

Note) For the services which require some basic applications or commands to be synchronized, such as acquisition of response from users to a specific instruction, it is assumed that synchronization is executed by the system on the roadside. The specific method of synchronization will be indicated in the process of the operation and therefore is out of stipulated range of this guideline.

2.4 Local Port Number of Basic Application Interface

For the local port number of each primitive function, the area 0x0C00 to 0x0C1F is used. The area is classified into four types after focusing the flow of information. Table 2.4-1 lists example of local port number for basic primitive application functions.

Table 2.4-1 Local port number of basic primitive application functions (example)

Port number	Basic primitive application functions	Data flow
0x0C00	OBE ID communication	Road to vehicle
0x0C01 - 0x0C07	For future use	
0x0C08	OBE basic instruction	Road to vehicle
0x0C09	OBE instruction response	
0x0C0A	Push-type information delivery	
0x0C0B - 0x0C0F	For future use	
0x0C10	IC card access	Road to vehicle /Vehicle to road, Use of card
0x0C11	Contactless IC smart card access	
0x0C12 - 0x0C17	For future use	
0x0C18	OBE memory access	Road to vehicle /Vehicle to road, Use of memory
0x0C19 - 0x0C1F	For future use	

Note) For the local port number for use of security platform, see Annex B. Load of applications on the remote station is judged by exchanging the local port number. Figure 2.4.1 shows the flow of the initial connection.

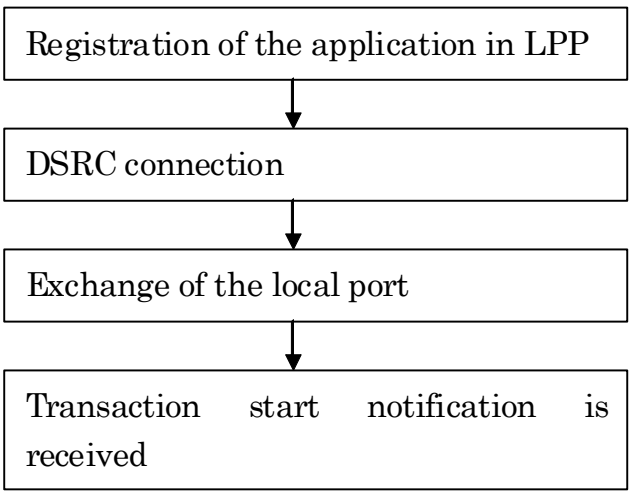


Figure 2.4.1 Flow of initial connection

[BLANK]

Chapter 3 Basic Application Interfaces Specification

3.1 OBE Instruction Response Application

3.1.1 Function Overview

The OBE instruction response application notifies the OBE of specific instruction information from the external server connected to the roadside system, and returns responses of the user to the roadside system using the input mechanism (such as buttons) provided on the OBE.

This application concretely provides the following two functions:

(1) In-vehicle instruction function to output the fee/toll information, etc. in voice from the OBE.

(2) In-vehicle response confirmation function to perform confirmation in the OBE through button pressing, voice output, etc.

3.1.2 Command

3.1.2.1 Command System

Commands (Obu Indication Command) used in the OBE instruction response application consist of normal commands and the denial response command from the OBE.

The normal commands consist of “OBE instruction notice command and OBE confirmation request command” from the roadside system to the OBE as well as the “OBE instruction response command and OBE confirmation response command” from the OBE to the roadside system.

The version number is “1” for the OBE instruction response application having the specifications described in this document.

3.1.2.2 Command Format

3.1.2.2.1 Normal Command

3.1.2.2.1.1 OBE Instruction Notice Command

The OBE instruction notice command is used when the roadside system notifies the OBE of the basic instruction information. Table 3-1-1 shows the command format.

Table3.1-1 OBE Instruction Notice Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type indicationRequest (0)							
4	Security Profile plainText(0)							
5	The data length of Operation Command Body ”opCommandBody” (10)							
6-15	The content of Operation CommandBody ”opCommandBody” OBE instruction notice information “Indication” format parameter							

(1) version

This field stores the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “indicationRequest(0)” to indicate the OBE instruction notice command.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) The data length of Operation Command Body ”opCommandBody”

This field indicates the data length (10) of succeeding operation command body. The unit is octet.

(b) The content of Operation Command Body ”opCommandBody”

This field is set to the OBE instruction information (“Indication” format, Refer to 3.1.3).

(i) transactionResult

This field indicates the communication result with the roadside system. Three values shown in Table 3.1-2 are defined, and other values are reserved For future use.

Table 3.1-2 Communication result format

Value	Meaning
0	The service is normally terminated without charge.
64	The service is abnormally terminated.
128	The service is normally terminated with charge.

(ii) time

This field is used to notify the time. Table 3.1-3 shows the format. “0x00 00 00 00” is given when there is no effective time information.

Table 3.1-3 Time format

stored order	item	bit size	data type
1	year	6	INTEGER(0..63)
2	month	4	INTEGER(0..12)
3	day	5	INTEGER(0..31)
4	hour	5	INTEGER(0..23)
5	minute	6	INTEGER(0..59)
6	second	6	INTEGER(0..59)

Note: The year is expressed as relative year from 2000.

(iii) amount

This field is used to notify the fee/toll. Table 3.1-4 shows the format.

Table 3.1-4 Fee Format

stored order	item	bit size	data type
1	amount	24	INTEGER(-8,388,608..8,388,607)
2	unit	16	BCD(4)

Note: “0x0392” specified in ISO4217 is stored as the unit.

3.1.2.2.1.2 OBE Instruction Response Command

The OBE instruction response command is used when the OBE notifies the roadside system of the normal operation in response to the basic instruction notice command. Table 3.1-5 shows the command format.

Table 3.1-5 OBE Instruction Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type indicationResponse(128)							

4	Security Profile plainText(0)
5	The data length of Operation Command Body ”opCommandBody” (0)

(1) version

This field stores the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “indicationResponse(128)” to indicate the OBE instruction response command.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) The data length of Operation Command Body ”opCommandBody”

This field indicates the operation command body. In this command, this field indicates the data length (0) of operation command body. The unit is octet.

3.1.2.2.1.3 OBE Confirmation Request Command

The OBE confirmation request command is used when the roadside system confirms whether any input such as button pressing is given to the OBE. Table 3.1-6 shows the command format.

Table 3.1-6 OBE Confirmation Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type confirmationRequest(1)							
4	Security Profile plainText(0)							
5	The data length of Operation Command Body ”opCommandBody” (1)							
6	The content of Operation Command Body ”opCommandBody” Confirmation Second information “ConfirmationSec” format parameter							

(1) version

This field stores the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “confirmationRequest(1)” to indicate the OBE confirmation request command.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) The data length of Operation Command Body

This field indicates the data length (1) of succeeding operation command body. The unit is octet.

(b) The content of Operation Command Body

This field is set to the second information of confirmation ("ConfirmationSec" format, Refer to 3.1.3).

(i) sec

The OBE confirms whether various inputs are given after receiving this command. It stores the time after this command is received until any input is confirmed. Table 3.1-7 shows the format.

Table 3.1-7 Format of sec

Unit: second (Fixed to 1 byte, the value from 0 to 255)

3.1.2.2.1.4 OBE Confirmation Response Command

The OBE confirmation response command is used when the OBE sends a response to the roadside system so that the roadside system can confirm whether any input such as button pressing and voice input is given to the OBE. Table 3.1-8 shows the command format.

Table 3.1-8 OBE Confirmation Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type confirmationResponse(129)							
4	Security Profile plainText(0)							
5	The data length of Operation Command Body "opCommandBody" (1)							
6	The content of Operation Command Body "opCommandBody" Confirmation Result information "confirmationResult" format parameter							

(1) version

This field stores the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “confirmationResponse(129)” to indicate the OBE confirmation response command.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) The data length of Operation Command Body ”opCommandBody”

This field indicates the data length (1) of succeeding operation command body. The unit is octet.

(b) The content of Operation Command Body ”opCommandBody”

This field is set to the result information of confirmation (“ConfirmationResult” format, Refer to 3.1.3).

(i) result

It stores whether any input is given to the OBE within the number of seconds specified in the “confirmationRequest” command. Table 3.1-9 shows the format.

Table 3.1-9 Confirmation Result

Confirmation Result (fixed to 1 byte) 0: No input, 1: Input indicating the approval (yes), 2: Input indicating the denial (no)

3.1.2.2.1.5 OBE Denial Response Command

The OBE denial response command is used when the OBE notifies the roadside system of negative acknowledgement in response to the basic instruction notice command. Table 3.1-10 shows the command format.

Table 3.1-10 OBE Denial Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			

2	Command Type “obuDenailResponse(255)”
3	Status
4	The length of supplement information “supplementInfo”
5 :	The contents of supplement information “supplementInfo”

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “obuDenialResponse(255)” to indicate the denial response command.

(3) Status

This field is set to the reason of denial response. For details, refer to Table 3.1-11.

Table 3.1-11 Status Code

Status code	Description
0	Not Use
1	There are no confirmation input means
2-3	For future use
4	The version incompatible
5-15	For future use
16	Illegal command
17-254	For future use
255	Other OBE internal error

(4) Supplement Information

(a) The length of supplement information “supplementInfo”

This field is set to the length of successing supplement information. The unit is octet. When there are no supplement information (this case is default case), this filed is set to value “0”.

(b) The contents of supplement information “supplementInfo”

This field is set to free information (the maximum length is 127 octets) as supplement information. When the version number is not the same, this field is set

to own version (“versionIndex” parameter).

3.1.3 Definition of Parameter Types

```
ObuIndicationCommand::=SEQUENCE{
    versionIndex        Version
    accessCommand       IndicationCommand
}

Version::=SEQUENCE{
    version     INTEGER(0..15),
    fill        BIT STRING(SIZE(4))     -- value of encoding is set up 0
}

IndicationCommand::=CHOICE{
    dummy               [0]     NULL,               -- not use
    operationCommand    [1]     OperationCommand,
    dummy               [2-254] NULL,               -- For future use
    obuDenialResponse   [255]   ObuDenialResponse
}

OperationCommand::=SEQUENCE{
    opCommandType       OpCommandType,
    opSecurityProfile   OpSecurityProfile,
    opCommandBody       OCTET STRING
}

OpCommandType::=ENUMERATED{
    indicationRequest           (0),    -- OBE instruction notice command
  message
    confirmationRequest         (1),    -- OBE confirm request command message
    reservedFor future use      (2-127), -- For future use
    indicationResponse          (128),  -- OBE instruction response command
  message
    confirmationResponse        (129),  -- OBE confirm response command message
```

```
    reservedFor future use        (130-255) -- For future use
}

Indication ::=SEQUENCE{
    transactionResult    INTEGER(0...255),
    time                 OCTET STRING(SIZE(4)),
    amount               OCTET STRING(SIZE(5))
}

ConfirmationSec ::=SEQUENCE{
    sec          INTEGER(0...255)
}

ConfirmationResult ::=SEQUENCE{
    result       Result
}

Result ::=ENUMERATED{
    noInput                  (0),      --no input
    approval                 (1),      --input indicated the approval
    denial                   (2),      --input indicated the denial
    reservedFor future use   (3-255)   -- For future use
}

ObuDenialResponse::=SEQUENCE{
    status               INTEGER(0...255),                -- status code
    supplementInfo       OCTET STRING(SIZE(0...255))      -- supplement information
}
```

3.1.4 Relationship with Other Standards

The relationships with other DSRC related standards used this application are shown below.

Table3.1-12 Relationship with other DSRC related standards

	DSRC related standard	Content used in this application
1	DSRC standard	AID=18 (DSRC Application Sub Layer)
2	NCP of ASL	LPCP (Local Port Control Protocol)
3	port number of LPCP	0x0C09
4	transaction service	Unidirectional data-sending transaction service in two transaction service provided by LPP.

3.1.5 Communication Procedures

This subsection describes the procedures for giving instructions to the OBE and receiving confirmation result from the OBE using the OBE instruction response application.

(1) Giving instructions to the OBE

(a) The roadside system notifies the OBE of the OBE instruction information using the OBE instruction notice command.

(b) When receiving the OBE instruction notice command, the OBE refers to the OBE instruction information, and outputs the contents. When output is completed, the OBE sends the OBE instruction response command to the roadside system.

(c) When rejecting the OBE instruction notice command due to the contents of the OBE instruction information or the OBE status in the step (b), the OBE sends to the roadside system the OBE denial response command instead of the OBE instruction response command.

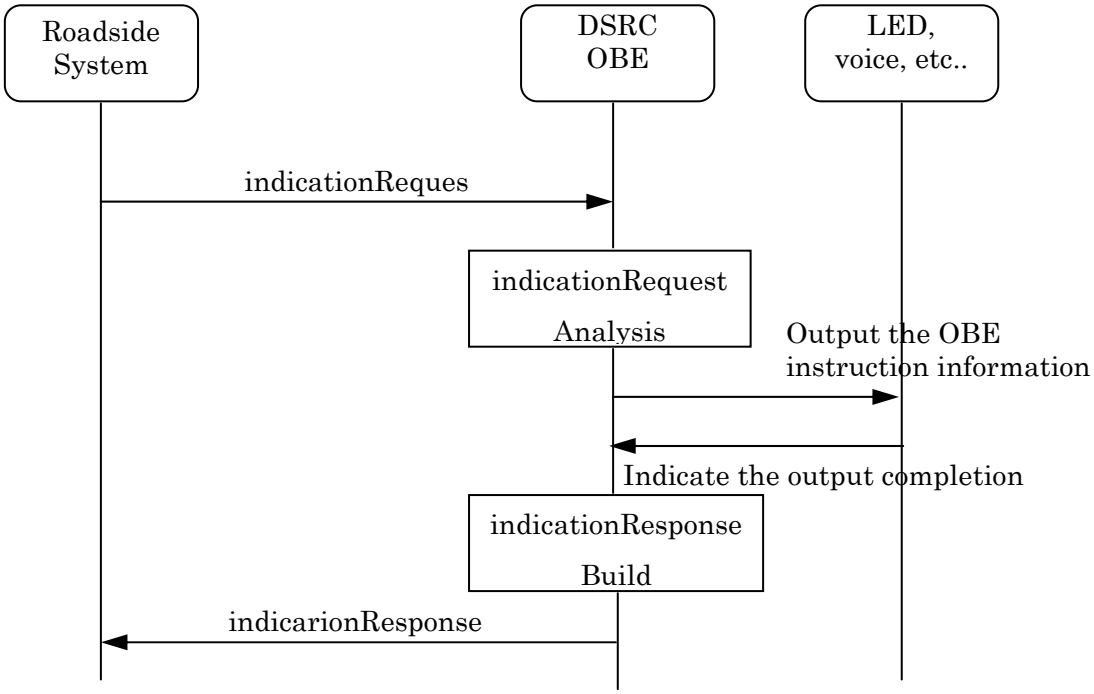


Figure 3.1-1 sequence example of instruction information notice to OBE

(2) Receiving confirmation result from the OBE

(a) The roadside system notifies the OBE of the OBE confirmation information using the OBE confirmation request command.

(b) When receiving the OBE confirmation request command, the OBE refers to the information on the number of seconds for confirmation, and waits for input of the confirmation information.

(c) When the confirmation information is input or is not input to the OBE within the number of seconds indicated in the information on the number of seconds for confirmation, the OBE notifies the roadside system of the confirmation result using the OBE confirmation response command.

(d) When the OBE does not have the confirmation information input means in the step (b), the OBE sends to the roadside system the OBE denial response command whose “status” indicates that “the OBE does not have confirmation input means”.

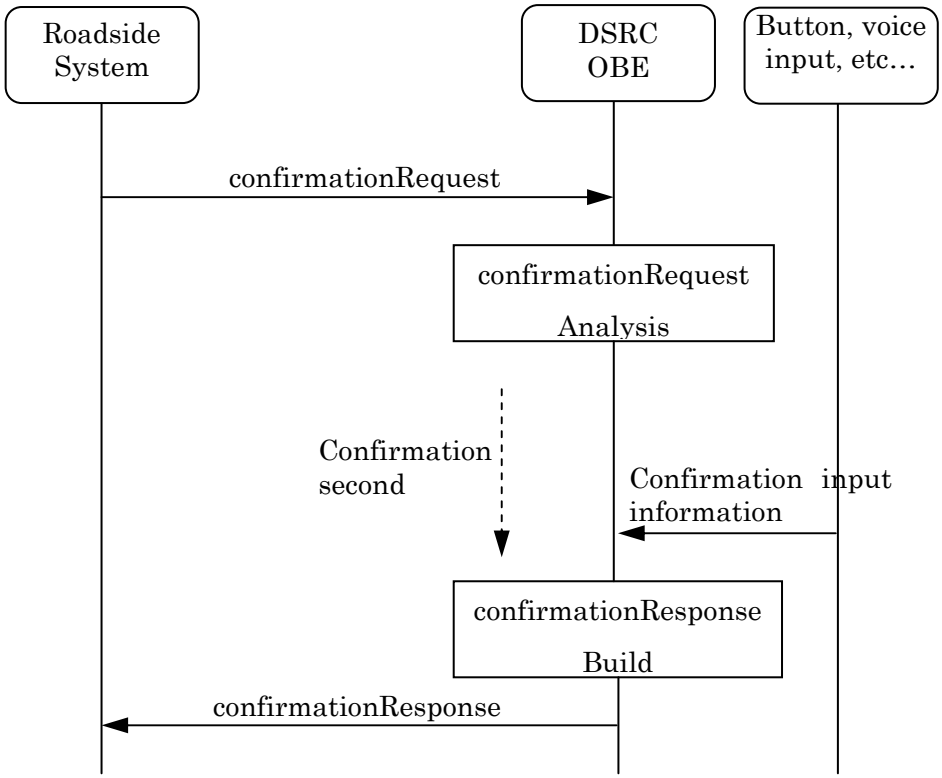


Figure 3.1-2 Sequence example of confirmation information notice to OBE

3.2 OBE Memory Access Application

3.2.1 Function Overview

The OBE memory access application reads or writes variable-length data stored in the memory inside the OBE in accordance with requests from the roadside system.

Data reading to and data writing from the memory inside the OBE from the roadside system shall be executed in units of data storage area specified and allocated in the roadside system or OBE. Each data storage memory area shall be identified by the identifier called memory tag.

The memory inside the OBE is mainly divided into the following two areas (memory control classification) by the data storage memory allocation method. The following control classification is specified explicitly in the 1st octet of the memory tag. (Refer to Annex.D for details.)

(1) Roadside system allocatable memory: Memory whose data storage area can be allocated from the roadside system

This memory becomes usable by the data storage memory allocation request from the roadside system or by memory allocation by the in-Vehicle system.

To the allocated data storage memory area, the roadside system can write and read variable-length data, and make the memory area free. The memory tag for the data storage memory area allocated in this memory is especially called “roadside system allocatable memory tag”. This memory is provided to allocate and use the memory for specific purpose inside the OBE after manufacturing of the OBE.

(2) OBE controlling memory: Memory whose data storage area cannot be allocated from the roadside system

This memory becomes usable when the OBE preliminarily allocates the data storage memory area and specifies the memory tag name corresponding to the data storage memory area.To the allocated data storage memory area, the roadside system can write and read variable-length data.

The memory tag for the data storage area allocated in this memory is especially called “OBE controlling memory tag”.

This memory is provided to read and write the data stored during manufacturing of the OBE (or subsequent registration work for setup, etc.), and exchange data between the roadside system and the external equipment connected to the OBE.

In addition to the above control classification, the memory inside the OBE is also classified into the following two areas by the record assurance method for the data storage memory. The following classification is specified explicitly in the 1st octet of the memory tag. (Refer to Annex.D for details.)

(1) Volatile memory area: Area in which the recorded information is not assured when the Vehicle engine is turned OFF or when the OBE power is turned OFF

(2) Nonvolatile memory area: Area in which the recorded information is assured even when the Vehicle engine is turned OFF or when the OBE power is turned OFF

The memory tag identifier shall consist of 8 octets.

This guideline defines each bit of the 1st octet for the memory control classification inside the OBE, record assurance, etc.

The 7-octet space from the 2nd octet to the 8th octet indicates the memory tag address, and is provided for unique setting for each system or provider. This guideline does not specify the assignment method of this 7-octet space. (Refer to Annex.D for details.)

The following attributes are set in each data storage memory area identified by the memory tag in the OBE. (Refer to Annex.D for details.)

(1) Protection mode

The protection mode is set for each memory tag (corresponding to each data storage memory area) against accesses from the roadside system.

The protection mode has three attributes, read permission, write permission and SPF (security platform) essentiality.

(2) Current data size

The current data size indicates the size of variable-length data stored in the data storage area.

(3) Memory allocation size

The memory allocation size (data storage upper limit) is set in the data storage memory area.

This set value can be set when the data storage memory area is allocated, and a different value can be set for each data storage memory area (corresponding to each memory tag).

Data storage up to the data size specified by this set value shall be assured in the OBE about writing from the roadside system.

When setting the memory allocation size in the data storage area, the communication resource (maximum data size for reading and writing from the roadside system) of the

OBE should be considered.

(4) Password (OPTIONAL)

Access restriction is set against accesses from the roadside system by the password authentication.

The password can be set when the data storage memory area is allocated, and different password can be set for each data storage memory area (corresponding to each memory tag). The OBE compares the registered password with the password set in various request commands from the roadside system, and then enables or disables accesses to the stored data.

When the password setting is handled as option, the OBE does not handle commands with password.

The OBE memory access application defined in this document has the following five functions:

(1) Function to inquire the use situation of the memory inside the OBE

(2) Function to allocate the data storage memory area in the memory inside the OBE (OPTIONAL)

(3) Function to make free the data storage memory area allocated in the memory inside the OBE (OPTIONAL)

(4) Function to read data from the memory inside the OBE

(5) Function to write data to the memory inside the OBE

Table 3.2-1 Available command types in each memory control classification

Memory Tag Type (Memory Tag)		OBE Controlling Memory (memory tag managed by OBE)			RSU Allocatable Memory (memory tag allocated by RSU)		
Protection Mode		RO	WO	R/W	RO	WO	R/W
(1)	OBE memory resource acquisition function	X			X		
(2)	Memory Alloc Function	-			X		
(3)	Memory Free Function	-			X		
(4)	Memory Read Function	X	-	X	X	-	X
(5)	Memory Write Function	-	X	X	-	X	X

Note: RO(ReadOnly) : Read Only Memory Tag
WO(WriteOnly) : Write Only Memory Tag
R/W(Read/Write) : Readable/Writable Memory Tag

3.2.1.1 OBE Memory Resource Information Acquisition Function

This function notifies the roadside system of the resource information on the OBE in response to a request from the roadside system. The roadside system can acquire the following information from the OBE:

(1) Maximum receivable data size : Maximum size of operation data which can be received by the OBE*

(2) Remaining capacity of the “roadside system allocatable memory” and number of “roadside system allocatable memory tags” which can be set (in both the volatile memory and the nonvolatile memory)

(3) Maximum number of memory tags which can be read or written at a time by batch processing

*Note:Maximum size (equivalent to the buffer size specified by “RegisterPort”) of the LPP-SDU subtracted by the header size of the OBE memory access application

The roadside system can acquire the attributes of data storage memory areas corresponding to memory tags by sending in advance the list of memory tags to be accessed using this function.

The OBE sends the attributes to the roadside system in response to an inquiry about memory tags already allocated and registered as data storage memory areas, and does not send the attributes of unregistered memory tags.

3.2.1.2 RSU Allocatable Memory Allocation Function (OPTIONAL)

This function allocates data storage memory areas inside the OBE, and registers memory tags (roadside system allocatable memory tags) specified as identifiers of data storage memory areas by the roadside system.

When the OBE cannot allocate new memory areas or when the roadside system points out an improper memory tag (OBE controlling memory tag), the OBE sends the OBE denial response command to the roadside system.

The OBE memory allocation function is classified into the following two types.

3.2.1.2.1 Memory Allocation (OPTIONAL)

This function allocates one data storage memory area inside the OBE, sets the attributes, and registers a memory tag based on the memory allocation information (such as memory tag, protection mode, memory allocation size and initial set value) excluding the password attribute

specified by the roadside system.

3.2.1.2.2 Memory Allocation with Password (OPTIONAL)

This function allocates one data storage memory area inside the OBE, sets the attributes, and registers a memory tag based on the memory allocation information (such as memory tag, protection mode, memory allocation size and initial set value) including the password attribute specified by the roadside system.

The difference from the memory allocation function described above is setting of the password as the attribute.

When the roadside system accesses a memory tag set by the "memory allocation with password" function, not only the memory tag but also the password is required.

3.2.1.3 RSU Allocatable Memory Free Function (OPTIONAL)

This function makes free one data storage memory area (roadside system allocable memory) allocated by the memory allocation function in accordance with a request from the roadside system.

When judging the area free request as proper based on the memory tag and attributes requested by the roadside system, the OBE deletes the registered memory tag, deletes data stored in the corresponding data storage memory area, makes free the corresponding data storage memory area, and sends the result to the roadside system.

When judging the area free request as improper because the request specifies an OBE controlling memory tag (OBE controlling memory), the OBE sends the OBE denial response command to the roadside system.

The OBE memory free function is classified into the following two types.

3.2.1.3.1 Memory Free (OPTIONAL)

This function makes free one data storage memory area inside the OBE based on the memory tag information specified by the roadside system.

This function is valid only to data storage memory areas (roadside system allocatable memory tags) allocated in the roadside system allocatable memory. When the roadside system specifies an OBE controlling memory tag, the OBE sends the OBE denial response command to the roadside system.

3.2.1.3.2 Memory Free with Password(OPTIONAL)

This function makes free one data storage memory area inside the OBE based on the

password attribute and memory tag information specified by the roadside system. The difference from the memory free function described above is that the OBE confirms agreement of the password before making the memory free. When the password disagrees, the OBE sends the OBE denial response command to the roadside system.

3.2.1.4 Memory Read Function

This function reads data stored in data storage memory areas in response to a request from the roadside system. When judging the memory read request as proper based on the memory tags and attributes requested by the roadside system, the OBE sends the stored data corresponding to the memory tags to the roadside system. When judging the memory read request as improper, the OBE sends the OBE denial response command to the roadside system. The memory reading function is classified into the following four types.

3.2.1.4.1 Memory Read

This function reads data stored in one data storage memory area not having the password attribute. The data size per command is restricted by the roadside system and the resource of the OBE. When the memory tag specified by the roadside system is not registered, the OBE sends the OBE denial response command to the roadside system.

3.2.1.4.2 Bulk Memory Read

This function reads data stored in one or more data storage memory areas not having the password attribute. The data size per command is restricted by the roadside system and the resource of the OBE. When an unregistered memory tag is included in memory tags specified by the roadside system, the OBE sends only the stored data corresponding to registered memory tags to the roadside system.

3.2.1.4.3 Memory Read with Password (OPTIONAL)

This function reads data stored in one data storage memory area having the password attribute. The data size per command is restricted by the roadside system and the resource of the OBE. When the memory tag specified by the roadside system is not registered or when the password disagrees, the OBE sends the OBE denial response command to the roadside system.

3.2.1.4.4 Bulk Memory Read with Password (OPTIONAL)

This function reads data stored in one or more data storage memory areas having the password attribute. The data size per command is restricted by the roadside system and the

resource of the OBE. When an unregistered memory tag or memory tag whose password disagrees is included in memory tags specified by the roadside system, the OBE sends only the stored data corresponding to registered memory tags whose password agrees to the roadside system.

3.2.1.5 Memory Write Function

This function writes data specified by the roadside system to data storage memory areas specified by memory tags in accordance with a request from the roadside system.

When judging the memory write request as proper based on the memory tags and attributes requested by the roadside system, the OBE writes corresponding stored data to data storage memory areas, and sends the writing result to the roadside system.

When judging the memory write request as improper, the OBE sends the OBE denial response command to the roadside system.

The memory writing function is classified into the following four types.

3.2.1.5.1 Memory Write

This function writes data specified by the roadside system to one data storage memory area not having the password attribute. The data size per command is not specified, but is restricted by the roadside system and the resource of the OBE. When the memory tag specified by the roadside system is not registered, the OBE sends the OBE denial response command to the roadside system.

3.2.1.5.2 Bulk Memory Write

This function writes data specified by the roadside system to one or more data storage memory areas not having the password attribute. The data size per command is not specified, but is restricted by the roadside system and the resource of the OBE. When an unregistered memory tag is included in memory tags specified by the roadside system, the OBE writes the specified data only to data storage memory areas corresponding to registered memory tags, and sends memory tags whose writing is successfully completed to the roadside system.

3.2.1.5.3 Memory Write with Password (OPTIONAL)

This function writes data specified by the roadside system to one data storage memory area having the password attribute. The data size per command is not specified, but is restricted by the roadside system and the resource of the OBE. When the memory tag specified by the roadside system is not registered or when the password disagrees, the OBE sends the OBE

denial response command to the roadside system.

3.2.1.5.4 Bulk Memory Write with Password (OPTIONAL)

This function writes data specified by the roadside system to one or more data storage memory areas having the password attribute. The data size per command is not specified, but is restricted by the roadside system and the resource of the OBE. When an unregistered memory tag or memory tag whose password disagrees is included in memory tags specified by the roadside system, the OBE writes the specified data only to data storage memory areas corresponding to registered memory tags whose password agrees, and sends memory tags whose writing is successfully completed to the roadside system.

3.2.2 Command

3.2.2.1 Command System

Commands in the OBE memory access application consist of normal commands and the OBE denial response command sent from the OBE. There are 26 normal commands as follows. The version number is “1” for the OBE memory access application having the specification described in this document.

- Memory Resource Information Acquisition Request Command
- Memory Resource Information Acquisition Response Command
- Memory Allocation Request Command (OPTIONAL)
- Memory Allocation Response Command (OPTIONAL)
- Memory Free Request Command (OPTIONAL)
- Memory Free Response Command (OPTIONAL)
- Memory Read Request Command
- Memory Read Response Command
- Memory Write Request Command
- Memory Write Response Command
- Bulk Memory Read Request Command
- Bulk Memory Read Response Command
- Bulk Memory Write Request Command
- Bulk Memory Write Response Command
- Memory Allocation with Password Request Command (OPTIONAL)
- Memory Allocation with Password Response Command (OPTIONAL)
- Memory Free with Password Request Command (OPTIONAL)
- Memory Free with Password Response Command (OPTIONAL)
- Memory Read with Password Request Command (OPTIONAL)
- Memory Read with Password Response Command (OPTIONAL)
- Memory Write with Password Request Command (OPTIONAL)
- Memory Write with Password Response Command (OPTIONAL)
- Bulk Memory Read with Password Request Command (OPTIONAL)
- Bulk Memory Read with Password Response Command (OPTIONAL)
- Bulk Memory Write with Password Request Command (OPTIONAL)
- Bulk Memory Write with Password Response Command (OPTIONAL)

3.2.2.2 Command Format

3.2.2.2.1 Normal Command

3.2.2.2.1.1 Memory Resource Information Acquisition Request Command

The memory resource information acquisition request command is used when the roadside system acquires from the OBE the information on the resource of the OBE and memory tags to be accessed subsequently. The OBE gives the information on specified memory tags using the memory resource information acquisition request command (resourceInfoResponse). Table 3.2-2 shows the command format.

Table 3.2-2 Memory Resource Information Acquisition Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type resourceInfoRequest(0)							
4	Security Profile plainText(0)							
5	Length of opCommandBody							
:	Content of opCommandBody “MemTagList” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “resourceInfoRequest(0)” to indicate the request command of memory resource information acquisition function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to the list of reading memory tag (“memTagList” format, Refer to Table 3.2-3). Maximum number of memory tags handled in this command is 30 or less.

Table 3.2-3 Memory Tag List (memTagList) Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	The number of memory tags							
:	1st memory tag memTag							
	:							
	Nth memory tag memTag							

(1) The number of memory tags

This field is set to the number of succeeding memory tags.

(2) Memory tag

This field indicates memory tag (“memTag” format parameter)

3.2.2.2.1.2 Memory Resource Information Acquisition Response Command

The memory resource information acquisition response command is used when the OBE notifies the roadside system of the information on resource of the OBE and memory tags specified by the roadside system. Table 3.2-4 shows the command format.

Table 3.2-4 Memory Resource Information Acquisition Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type resourceInfoResponse(128)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody “resourceInfo” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “resourceInfoResponse(128)” to indicate the response command of memory resource information acquisition function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to the resource information of memory tag (“resourceInfo” format, Refer to Table 3.2-5).

Table 3.2-5 The Resource Information of Memory Tag (resourceInfo) Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1-4	Max Receivable Data Size maxCommandBodySize							
5-16	Property of Memory Area storageProperty							
	Remaining number of memory tags in the nonvolatile memory area availableNonVolatileTagNum							
	Remaining capacity of data storage areas in the nonvolatile memory area availableNonVolatileDataCapacity							
	Remaining number of memory tags in the volatile memory area availableVolatileTagNum							
	Remaining capacity of data storage areas in the volatile memory area availableVolatileDataCapacity							
17-18	Maximum number of memory tags handled in batch processing bulkTagNum							
19 :	Memory tag resource information list “tagResourceList” format parameter							

(1) Maximum Receivable Data Size

This field indicates the maximum operation data size which can be received by the OBE.

This field is set to "4294967295” when the maximum receivable data size exceeds 4294967295 bytes.

(2) Property of Memory Area

This field indicates the following areas when the OBE incorporates the memory allocation function (roadside system allocable memory).

This field is set to “0” to all of the following areas for the OBE not incorporating the

memory allocation function (roadside system allocatable memory).

(a) Remaining number of memory tags in the nonvolatile memory area

This field indicates the remaining number of memory tags available in the nonvolatile memory area in the roadside system allocatable memory.

This field is set to “65535” when the number of allocatable memory tags exceeds 65535.

This field is set to “0” for the OBE not incorporating the nonvolatile memory area.

(b) Remaining capacity of data storage areas in the nonvolatile memory area

This field indicates the remaining capacity of nonvolatile memory area in the roadside system allocatable memory.

This field is set to “4294967295” when the remaining capacity exceeds 4294967295 bytes. This field is set to “0” for the OBE not incorporating the nonvolatile memory area.

(c) Remaining number of memory tags in the volatile memory area

This field indicates the remaining number of memory tags available in the volatile memory area in the roadside system allocatable memory.

This field is set to “65535” when the number of allocatable memory tags exceeds 65535.

This field is set to “0” for the OBE not incorporating the volatile memory area.

(d) Remaining capacity of data storage areas in the volatile memory area

This field indicates the remaining capacity of volatile memory area in the roadside system allocatable memory.

This field is set to “4294967295” when the remaining capacity exceeds 4294967295 bytes. This field is set to “0” for the OBE not incorporating the volatile memory area.

(3) Maximum number of memory tags handled in batch processing

This field indicates the maximum number of memory tags in the OBE which can be read or written in batch processing.

(4) Memory tag resource information list

Memory tags and attributes (“tagResourceList” type shown in Table 3.2-6) registered (whose memory area is allocated) in the OBE is stored here among memory tags shown in the memory tag list (“memTagList” format parameter) contained in “resourceInfoRequest”.

Table 3.2-6 The List of Resource Information of Memory Tag (tagResourceList) Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	The number of memory tags							
:	Resource information of 1st memory tag tagResourceInfo							
	:							
	Resource information of Nth memory tag tagResourceInfo							

(1) The number of memory tags

This field is set to the number of succeeding “tagResourceInfo”.

(2) Resource Information of Memory Tag

This field is set to resource information of memory tag(“tagResourceInfo” format, Refer to Table 3.2-7).

Table 3.2-7 Resource Information of Memory Tag (tagResourceInfo) Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1-8	Memory Tag memTag							
9	The Attribute of Memory Tag tagAttribute							
:	fill(0)					Protection Mode permission		
						spf	WritePro tect	readProt ect
	The Size of Memory Tag Data tagDataSize							
	The Max Data Size of Memory Tag Data maxMemorySize							

(1) Memory Tag

This field is set to memory tag.

(2) Protection Mode

This field indicates the protection mode of the data storage memory area identified by the memory tag. The protection mode is represented by 3 bits indicating “essential/nonessential” or “enabled/disabled”.

(a) spf

This field is set to "1” to make essential the use of the SPF for accessing the data storage memory area.

(b) writeProtect

This field is set to “1” to disable writing to the data storage memory area.

(c) readProtect

This field is set to “1” to disable reading from the data storage memory area.

(3) The Size of Memory Tag Data

This field indicates the size of data currently in the data storage memory area identified by the memory tag. The field is set to “0” to a memory tag for which reading is disabled in the protection mode.

(4) The Max Data Size of Memory Tag Data

This field indicates the capacity of the data storage memory area identified by the memory tag. This field is set to "4294967295” when the allocated memory size exceeds 4294967295 bytes.

3.2.2.2.1.3 Memory Allocation Request Command

The memory allocation request command is used when the roadside system allocates the roadside system allocatable memory in the OBE. The OBE gives response using the memory allocation response command (memoryAllocResponse). Table 3.2-8 shows the command format.

Table 3.2-8 Memory Allocation Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type memoryAllocRequest (1)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody “memoryAllocInfo” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “memoryAllocRequest (1)” to indicate the request command of memory allocation function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This filed is set to the memory allocation information (“memoryAllocInfo” format, Refer to Table 3.2-9).

Table 3.2-9 Memory Allocation Information (memoryAllocInfo) Format

<table>
<tr><td></td><td>7(MSB)</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0(LSB)</td></tr>
<tr><td>1-8</td><td colspan="8">Memory Tag
memTag</td></tr>
<tr><td rowspan="4">9
:</td><td colspan="5" rowspan="2">fill(0)</td><td colspan="3">Protection Mode
permission</td></tr>
<tr><td>spf</td><td>writeProt
ect</td><td>readProt
ect</td></tr>
<tr><td colspan="8">Memory Allocation Size
maxMemorySize</td></tr>
<tr><td colspan="8">Initial Set Value
initialValue</td></tr>
</table>

(1) Memory tag

The memory tag name corresponding to the data storage memory area to be allocated is stored here.

(2) Protection mode

This field indicates the protection mode of the data storage memory area to be allocated.

The protection mode is represented by 3 bits indicating “essential/nonessential” or “enabled/disabled”.

(a) spf

This field is set to "1” to make essential the use of the SPF for accessing the data storage memory area.

(b) writeProtect

This field is set to “1” to disable writing to the data storage memory area.

(c) readProtect

This field is set to “1” to disable reading from the data storage memory area.

(3) Memory allocation size

This field indicates the memory capacity to be allocated under consideration of the communication resource (maximum data size which can be read and written from the roadside system) of the OBE.

(4) Initial set value

This field indicates the data initial value to be stored when the data storage memory

area is allocated.

3.2.2.2.1.4 Memory Allocation Response Command

The memory allocation response command is used when the OBE notifies the roadside system that memory allocation requested by the memory allocation request command is normally finished. Table 3.2-10 shows the command format.

Table 3.2-10 Memory Allocation Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type memoryAllocResponse(129)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
6-13	The content of OpCommandBody “memTag” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “memoryAllocResponse (129)” to indicate the response command of memory allocation function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to memory tag (“memTag” format parameter).

3.2.2.3 Memory Free Request Command

The memory free request command is used when the roadside system makes free the roadside system allocatable memory already allocated in the OBE.

The OBE gives response using the memory free response command (memoryFreeResponse). Table 3.2-11 shows the command format.

Table 3.2-11 Memory Free Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type memoryFreeRequest (2)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody “memTag” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “memoryFreeRequest (2)” to indicate the request command of memory free function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This filed is set to memory tag (“memTag” format parameter) .

3.2.2.4 Memory Free Response Command

This command is used when the OBE notifies the roadside system that memory free requested by the memory free request command is normally finished. Table 3.2-12 shows the command format.

Table 3.2-12 Memory Free Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type memoryFreeResponse(130)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody(0)							
6-13	The content of OpCommandBody “memTag” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “memoryFreeResponse(130)” to indicate the response command of memory free function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This filed is set to memory tag (“memTag” format parameter) .

3.2.2.5 Memory Read Request Command

The memory read request command is used when the roadside system reads one data storage memory area (one memory tag) in the OBE. The OBE gives response about the data storage memory area specified by the roadside system using the memory read response command (readResponse). Table 3.2-13 shows the command format.

Table 3.2-13 Memory Read Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type readRequest(3)							
4	Security Profile plainText(0)							
5-13	The data length of OpCommandBody							
	The content of OpCommandBody “memTag” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “readRequest(3)” to indicate the request command of memory read function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field is set to the data length(8) of succeeding operation command body. The unit is octet.

(b) This filed is set to memory tag (“memTag” format parameter).

3.2.2.6 Memory Read Response Command

The memory read response command is sent from the OBE to the roadside system as response when the roadside system reads one data storage memory area (one memory tag) in the OBE. Table 3.2-14 shows the command format.

Table 3.2-14 Memory Read Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type readResponse(131)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody “memData” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “readResponse(131)” to indicate the response command of memory read function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This filed is set to memory data information (“memData” format parameter, Refer to Table 3.2-15).

Table 3.2-15 The Memory Data Information (memData) Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1-8	Memory Tag memTag							
9 :	The length of Stored Data							
	The contents of Stored Data							

(1) Memory Tag

This field is set to memory tag.

(2) Stored Data

(a) This field is set to the data length of succeeding stored data.

(b) This field is set to the store data coresspond to memory tag.

3.2.2.7 Memory Write Request Command

The memory write request command is used when the roadside system writes specified data to one data storage memory area corresponding to one memory tag in the OBE. The OBE gives response using the memory write response command (writeResponse) when the write request is normally executed. Table 3.2-16 shows the command format.

Table 3.2-16 Memory Write Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type writeRequest(4)							

4	Security Profile plainText(0)
5 :	The data length of OpCommandBody
	The content of OpCommandBody “memData” format parameter

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “writeRequest (4)” to indicate the request command of memory write function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This filed is set to memory data information (“memData” format parameter, Refer to Table 3.2-15).

3.2.2.8 Memory Write Response Command

The memory write response command is sent from the OBE to the roadside system as response when the roadside system writes specified data to one data storage area corresponding to one memory tag in the OBE. Table 3.2-17 shows the command format.

Table 3.2-17 Memory Write Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							

3	Operation Type writeResponse(132)
4	Security Profile plainText(0)
5	The data length of OpCommandBody
6-13	The content of OpCommandBody “memTag” format parameter

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “writeResponse(132)” to indicate the response command of memory write function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This filed is set to writing memory tag (“memTag” format parameter).

3.2.2.9 Bulk Memory Read Request Command

The bulk memory read request command is used when the roadside system reads specified one or more data storage memory areas in the OBE. The OBE gives response about one or more data storage memory areas specified by the roadside system using the bulk memory read response command (readBulkResponse). Table 3.2-18 shows the command format.

Table 3.2-18 Bulk Memory Read Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			

2	Command Type operationCommand(1)
3	Operation Type readBulkRequest(5)
4	Security Profile plainText(0)
5	The data length of OpCommandBody
:	The content of OpCommandBody “memTagList” format parameter

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “readBulkRequest (5)” to indicate the request command of bulk memory read function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This filed is set to memory tag list information (“memTagList” format parameter, Refer to Table 3.2-19).

Table 3.2-19 Memory Tag List Information (memTagList) Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	The number of Memory Tag							
:	1st Memory Tag memTag							
	:							
	:							

	Nth Memory Tag memTag

(1) The number of Memory Tags

This field is set to the number of succeeding “memTag”.

(2) Memory Tag

This filed is set to memory tag (“memTag” format parameter).

3.2.2.10 Bulk Memory Read Response Command

The bulk memory read response command is used when the OBE sends to the roadside system one or more data stored in one or more data storage memory areas specified by the roadside system.Table 3.2-20 shows the command format.

Table 3.2-20 Bulk Memory Read Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type readBulkResponse(133)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody “memDataList” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “readBulkResponse (133)” to indicate the response command of bulk memory read function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to the data list information ("memDataList" format parameter, Refer to Table 3.2-21).

Table 3.2-21 Data List Information (memDataList) Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	The number of Memory Data Information							
:	1st Memory Data Information memData							
	:							
	Nth Memory Data Information memData							

(1) The number of Memory Data Information

This field is set to the number of succeeding "memData".

(2) Memory Data Information

This field is set to the memory data information ("memData" format, Refer to Table 3.2-15).

3.2.2.11 Bulk Memory Write Request Command

The bulk memory write request command is used when the roadside system writes one or more data to one or more data storage memory areas allocated in the OBE. The OBE gives response about memory tags whose data writing is successfully finished among specified one or more memory tags using the bulk memory write response command (writeBulkResponse). Table 3.2-22 shows the command format.

Table 3.2-22 Bulk Memory Write Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)

1	version	fill(0)
2	Command Type operationCommand(1)	
3	Operation Type writeBulkRequest(6)	
4	Security Profile plainText(0)	
5	The data length of OpCommandBody	
:	The content of OpCommandBody “memDataList” format parameter	

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “wirteBulkRequest (6)” to indicate request command of the bulk memory write function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to the writing information list (“memDataList” format parameter, Refer to Table 3.2-21).

3.2.2.12 Bulk Memory Write Response Command

The bulk memory write response command is sent from the OBE to the roadside system as response when the roadside system writes one or more data at one time to the OBE. The OBE gives response about memory tags whose data writing is successfully finished among specified one or more memory tags. Table 3.2-23 shows the command format.

Table 3.2-23 Bulk Memory Write Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type writeBulkResponse(134)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody “memTagList” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “wirteBulkResponse (134)” to indicate the response command of bulk memory write function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to the memory tag information list (“memTagList” format parameter, Refer to Table 3.2-3).

3.2.2.13 Memory Allocation with Password Request Command

The memory allocation with password request command is used when the roadside system allocates the roadside system allocatable memory in the OBE and set password attribute. The OBE gives response using the memory allocation with password response command

(memoryAllocResponseWithCredence). Table 3.2-24 shows the command format.

Table 3.2-24 Memory Allocation with Password Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type memoryAllocRequestWithCredence (65)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody “memoryAllocInfoWithCredence” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “memoryAllocRequestWithCredence (65)” to indicate the request command of memory allocation with password function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to the memory allocation information with password (“memoryAllocInfoWithCredence” format parameter, Refer to Table 3.2-25).

Table 3.2-25 The Memory Allocation Information with Password (memoryAllocInfoWithCredence) Format

<table>
<tr><th></th><th>7(MSB)</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0(LSB)</th></tr>
<tr><td>1-8</td><td colspan="8">Memory Tag
memTag</td></tr>
<tr><td rowspan="5">9
:</td><td colspan="5" rowspan="2">fill(0)</td><td colspan="3">Protection Mode
permission</td></tr>
<tr><td>spf</td><td>writeProt
ect</td><td>readProt
ect</td></tr>
<tr><td colspan="8">Memory Allocation Size
maxMemorySize</td></tr>
<tr><td colspan="8">Password
accessCredential</td></tr>
<tr><td colspan="8">Initial Set Value
initialValue</td></tr>
</table>

(1) Memory tag

The memory tag name corresponding to the data storage memory area to be allocated is stored here.

(2) Protection mode

This field indicates the protection mode of the data storage memory area to be allocated. The protection mode is represented by 3 bits indicating “essential/nonessential” or “enabled/disabled”.

(a) spf

This field is set to "1” to make essential the use of the SPF for accessing the data storage memory area.

(b) writeProtect

This field is set to “1” to disable writing to the data storage memory area.

(c) readProtect

This field is set to “1” to disable reading from the data storage memory area.

(3) Memory allocation size

This field indicates the memory capacity to be allocated under consideration of the communication resource (maximum data size which can be read and written from the roadside system) of the OBE.

(4) Password

This field indicates password (8 octet).

(5) Initial set value

This field indicates the data initial value to be stored when the data storage memory area is allocated.

3.2.2.14 Memory Allocation with Password Response Command

The memory allocation with password response command is used when the OBE notifies the roadside system that memory allocation requested by the memory allocation with password request command is normally finished. Table 3.2-26 shows the command format.

Table 3.2-26 Memory Allocation with Password Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type memoryAllocResponseWithCredence(193)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
6-13	The content of OpCommandBody “memTag” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “memoryAllocResponseWithCredence (193)” to indicate the response command of memory allocation with password function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to memory tag information allocated successfully (“memTag” format parameter).

3.2.2.15 Memory Free with Password Request Command

The memory free with password request command is used when the roadside system makes free the roadside system allocatable memory already allocated in the OBE. The OBE gives response using the memory free with password response command (memoryFreeResponseWithCredence). Table 3.2-27 shows the command format.

Table 3.2-27 Memory Free with Password Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type memoryFreeRequestWithCredence (66)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody “memTagWithCredence” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “memoryFreeRequestWithCredence (66)” to indicate the request command of memory free with password function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this

command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to memory tag with password information (“memTagWithCredence” format parameter, Refer to Table 3.2-28).

Table 3.2-28 Memory Tag with Password Information (memTagWithCredence) Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1-8	Memory Tag memTag							
9-16	Password accessCredential							

(1) Memory Tag

This field is set to memory tag.

(2) Password

This field is set to password (8 Bytes) .

3.2.2.16 Memory Free with Password Response Command

This command is used when the OBE notifies the roadside system that memory free requested by the memory free with password request command is normally finished. Table 3.2-29 shows the command format.

Table 3.2-29 Memory Free with Password Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type memoryFreeResponseWithCredence(194)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							

6-13	The content of OpCommandBody “memTag” format parameter

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “memoryFreeResponseWithCredence(194)” to indicate the response command of memory free with password function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to memory tag information processed successfully (“memTag’” format parameter).

3.2.2.17 Memory Read with Password Request Command

The memory read with password request command is used when the roadside system reads one data storage memory area (one memory tag) with password in the OBE. The OBE gives response about the data storage memory area specified by the roadside system using the memory read with response command (readResponseWithCredence). Table 3.2-30 shows the command format.

Table 3.2-30 Memory Read with Password Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type readRequestWithCredence(67)							

4	Security Profile plainText(0)
5	The data length of OpCommandBody
:	The content of OpCommandBody “memTagWithCredence” format parameter

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “readRequestWithCredence(67)” to indicate the request command of memory read with password function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This filed is set to memory tag with password information (“memTagWithCredence” format parameter, Refer to Table 3.2-28).

3.2.2.18 Memory Read with Password Response Command

The memory read with password response command is sent from the OBE to the roadside system as response when the roadside system reads one data storage memory area (one memory tag) restricted to read by password in the OBE.

Table 3.2-31 shows the command format.

Table 3.2-31 Memory Read with Password Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							

3	Operation Type readResponseWithCredence(195)
4	Security Profile plainText(0)
5 :	The data length of OpCommandBody
	The content of OpCommandBody “memData” format parameter

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “readResponseWithCredence(195)” to indicate the response command of memory read with password function

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to stored memory data information (“memData” format parameter, Refer to Table 3.2-15).

3.2.2.19 Memory Write with Password Request Command

The memory write with password request command is used when the roadside system writes specified data to one data storage memory area corresponding to one memory tag with password in the OBE. The OBE gives response using the memory write with password response command (writeResponseWithCredence) when the write request is normally executed. Table 3.2-32 shows the command format.

Table 3.2-32 Memory Write with Password Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)

1	version	fill(0)
2	Command Type operationCommand(1)	
3	Operation Type writeRequestWithCredence(68)	
4	Security Profile plainText(0)	
5	The data length of OpCommandBody	
:	The content of OpCommandBody “memDataWithCredence” format parameter	

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “writeRequestWithCredence (68)” to indicate the response command of memory write with password function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to writing memory data information with password (“memDataWithCredence” format parameter, Refer to Table 3.2-33).

Table 3.2-33 Memory Data Information with Password (memDataWithCredence) Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Memory Tag with Password “memTagWithCredence” format parameter							
:	The length of memory data							

	The contents of memory data

(1) Memory Tag with Password

This field is set to memory tag information with password (Refer to Table 3.2-28).

(2) Memory Data

(a) This field is set to the length of successing memory data.

(b) This field is set to data corresponding to memory tag information.

3.2.2.20 Memory Write with Password Response Command

The memory write with password response command is sent from the OBE to the roadside system as response when the roadside system writes specified data to one data storage area corresponding to one memory tag with password in the OBE. Table 3.2-34 shows the command format.

Table 3.2-34 Memory Write with Password Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type writeResponseWithCredence(196)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
6-13	The content of OpCommandBody “memTag” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “writeResponseWithCredence(196)” to indicate the

response command of memory write with password function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to memory tag information ("memTag" format parameter).

3.2.2.21 Bulk Memory Read with Password Request Command

The bulk memory read with password request command is used when the roadside system reads specified one or more data storage memory areas with password in the OBE. The OBE gives response about one or more data storage memory areas specified by the roadside system using the bulk memory read with password response command (readBulkResponseWithCredence). Table 3.2-35 shows the command format.

Table 3.2-35 Bulk Memory Read with Password Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type readBulkRequestWithCredence(69)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody "memTagListWithCredence" format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier "operationCommand(1)" to indicate the normal command.

(3) Operation Type

This field is set to an identifier “readBulkRequestWithCredence (69)” to indicate the request command of bulk memory read with password function.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to list of memory tags with password information (“memTagListWithCredence” format parameter, Refer to Table 3.2-36).

Table 3.2-36 List of Memory Tags with Password Information (memTagListWithCredence) Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	The number of memory tags with password							
:	1st Memory Tag with Password “memTagWithCredence” format parameter							
	:							
	Nst Memory Tag with Password “memTagWithCredence” format parameter							

(1) The Number of Memory Tags with Password

This field is set to the number of successing memory tags with password.

(2) Memory Tag with Password

This filed is set to memory tag with password (“memTagWithCredence” format parameter, Refer to Table 3.2-28).

3.2.2.22 Bulk Memory Read with Password Response Command

The bulk memory read with password response command is used when the OBE sends to the roadside system one or more data stored in one or more data storage memory areas with password specified by the roadside system. Table 3.2-37 shows the command format.

Table 3.2-37 Bulk Memory Read with Password Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type readBulkResponseWithCredence(197)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody "memDataList"' format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier "operationCommand(1)" to indicate the normal command.

(3) Operation Type

This field is set to an identifier "readBulkResponseWithCredence (197)" to indicate the bulk memory read with password response.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to data list information ("memDataList" format parameter, Refer to Table 3.2-21).

3.2.2.23 Bulk Memory Write with Password Request Command

The bulk memory write with password request command is used when the roadside system writes one or more data to one or more data storage memory areas with password allocated in the OBE. The OBE gives response about memory tags whose data writing is successfully finished

among specified one or more memory tags using the bulk memory write with password response command (writeBulkResponseWithCredence). Table 3.2-38 shows the command format.

Table 3.2-38 Bulk Memory Write with Password Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type writeBulkRequestWithCredence(70)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody “memDataListWithCredence” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “wirteBulkRequestWithCreence (70)” to indicate the bulk memory write with password request.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to list of memory data with password (“memDataListWithCredence” format parameter, Refer to Table 3.2-39).

Table 3.2-39 List of Memory Data with Password (memDataListWithCredence) Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	The number of memory data with password							
:	1st memory data with password “memDataWithCredence” format parameter							
	: :							
	Nth memory data with password “memDataWithCredence” format parameter							

(1) The number of memory data with password

This field is set to the number of successing memory data with password.

(2) Memory Data with Password

This field is set to the memory data with password (“memDataWithCredence” format parameter, Refer to Table 3.2-33).

3.2.2.24 Bulk Memory Write with Password Response Command

The bulk memory write with password response command is sent from the OBE to the roadside system as response when the roadside system writes one or more data at one time to the OBE. The OBE gives response about memory tags whose data writing is successfully finished among specified one or more memory tags. Table 3.2-40 shows the command format.

Table 3.2-40 Bulk Memory Write with Password Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							

3	Operation Type writeBulkResponseWithCredence(198)
4	Security Profile plainText(0)
5 :	The data length of OpCommandBody
	The content of OpCommandBody “memTagList” format parameter

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “wirteBulkResponseWithCredence (198)” to indicate the bulk memory write with password response.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) This field indicates the data length of succeeding operation command body. The unit is octet.

(b) This field is set to the list of memory tag information written successfully (“memTagList” format parameter, Refer to Table 3.2-3).

3.2.2.25 OBU Denial Response Command

OBU denial response command is used when the OBE notifies the roadside system of negative acknowledgement. Table 3.2-41 shows the command format.

Table 3.2-41 OBU Denial Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type obuDenialResponse(255)							

3	status
4	The length of supplement information
5 :	The contents of supplement information

(1) version

This field indicates the application version.

(2) Command Type

This field is set to obuDenialRespnose(255) as OBU Denial Response Conmand.

(3) status

This field is set to the reason of denial response. For details, refer to (Table 3.2-42).

(4) Supplement Information

(a) The length of supplement information

This field is set to the length of successing supplement information. The unit is octet. When there are no supplement information (this case is default case), this filed is set to value “0”.

(b) The contents of supplement information

This field is set to free information (the maximum length is 127 octets) as supplement information. When version number is not same, this field is set to own version (“versionIndex” parameter).

Table 3.2-42 Status Code

Status Code	Description
0	For future use
1	Failure to Read from OBE Memory
2	Failure to Write to OBE Memory
3	Failure to Allocate/Free OBE Memory
4	The version incompatible
5	Memory Area is insufficient to allocate
6	There are no requested memory tag
7	Over maximum number of tags in bulk function
8	Protection mode violation
9	Access limit violation
10	The password incompatible

11	SPF violation
12	No support command
13-15	For future use
16	Illegal command
17-254	For future use
255	Other OBE internal error

3.2.3 Definition of ParameterTypes

```
ObuMemoryAccessCommand::=SEQUENCE{
    versionIndex        Version,
    accessCommand       AccessCommand
}

Version::=SEQUENCE{
    version             INTEGER(0...15),
    fill                BIT STRING(SIZE(4))     -- encoding value is 0
}

AccessCommand::=CHOICE{
    dummy               [0]     NULL,                   -- don't use
    operationCommand    [1]     OperationCommand,
    dummy               [2-254] NULL,                   -- For future use
    obuDenialResponse   [255]   ObuDenialResponse
}

OperationCommand::=SEQUENCE{
    opCommandType           OpCommandType,
    opSecurityProfile       OpSecurityProfile,
    opCommandBody           OCTET STRING
}

OpCommandType::=ENUMERATED{
    resourceInfoRequest     (0),-- Memory Resource Information Acquisition
  Request Command
    memoryAllocRequest      (1), -- Memory Allocation Request Command
    memoryFreeRequest       (2), -- Memory Free Request Command
    readRequest             (3), -- Memory Read Request Command
    writeRequest            (4), -- Memory Write Request Command
    readBulkRequest         (5), -- Bulk Memory Read Request Command
    writeBulkRequest        (6), -- Bulk Memory Write Request Command
```

reservedFor future use (7-64), -- For future use
memoryAllocRequestWithCredence (65),
--Memory Allocation with Password Request Command
memoryFreeRequestWithCredence (66),
-- Memory Free with Password Request Command
readRequestWithCredence (67),
-- Memory Read with Password Request Command
writeRequestWithCredence (68),
-- Memory Write with Password Request Command
readBulkRequestWithCredence (69),
-- Bulk Memory Read with Password Request Command
writeBulkRequestWithCredence (70),
-- Bulk Memory Write with Password Request Command
reservedFor future use (71-127), -- For future use
resourceInfoResponse (128),
-- Memory Resource Information Acquisition Response Command
memoryAllocResponse (129),
-- Memory Allocation Response Command
memoryFreeResponse (130),
-- Memory Free Response Command
readResponse (131),
-- Memory Read Response Command
writeResponse (132),
-- Memory Write Response Command
readBulkResponse (133),
-- Bulk Memory Read Response Command
writeBulkResponse (134),
-- Bulk Memory Write Response Command
reservedFor future use (135-192), -- For future use
memoryAllocResponseWithCredence (193),
-- Memory Allocation with Password Response Command
memoryFreeResponseWithCredence (194),
-- Memory Free with Password Response Command
readResponseWithCredence (195),
-- Memory Read with Password Response Command

```
    writeResponseWithCredence          (196),
        -- Memory Write with Password Response Command
    readBulkResponseWithCredence       (197),
        -- Bulk Memory Read with Password Response Command
    writeBulkResponseWithCredence      (198),
        -- Bulk Memory Write with Password Response Command
    reservedFor future use             (199-255) ,      -- For future use
}

OpSecurityProfile::=ENUMERATED{
    plainText                  (0),            -- plain text
    reservedFor future use     (1-255)         -- For future use
}

ObuDenialResponse::=SEQUENCE{
    status                 INTEGER(0...255),       -- status code
    supplementInfo         OCTET STRING(SIZE(0..255))
                                                   -- supplement information
}

ResourceInfo::= SEQUENCE{
    maxCommandBodySize         INTEGER(0...4294967295),
        -- Maximum Receivable Data Size
    storageProperty            StorageProperty,
        -- Property of Memory Area
    bulkTagNum                 INTEGER(0...65535),
        -- Maximum number of memory tags handled in batch processing
    tagResourceList            TagResourceList
        -- Memory tag resource information list
}

StorageProperty::=SEQUENCE{
    availableNonVolatileTagNum          INTEGER(0...65535),
        -- Remaining number of memory tags in the nonvolatile memory area
    availableNonVolatileDataCapacity    INTEGER(0...4294967295),
```

```
        -- Remaining capacity of data storage areas in the nonvolatile memory area
    availableVolatileTagNum          INTEGER(0...65535),
        -- Remaining number of memory tags in the volatile memory area
    availableVolatileDataCapacity    INTEGER(0...4294967295)
        -- Remaining capacity of data storage areas in the volatile memory area
}

TagResourceList::= SEQUENCE OF TagResourceInfo.

TagResourceInfo::= SEQUENCE{
    tag                 MemTag,         -- Memory Tag
    tagAttribute        TagAttribute    -- The Attribute of Memory Tag
}

TagAttribute::= SEQUENCE{
    fill                BITSTRING(SIZE(5)),
    permission          Permission,     -- Protection mode
    tagDataSize         INTEGER(0...4294967295),
                            -- The Size of Memory Tag Data
    maxMemorySize       INTEGER(0...4294967295)
                            -- The Max Data Size of Memory Tag Data
}

Permission::= SEQUENCE {
    spf                 BOOLEAN,        -- SPF usage case is 1
    writeProtect        BOOLEAN,        -- Write prohibit case is 1
    readProtect         BOOLEAN         -- Read prohibit case is 1
}

MemoryAllocInfo ::= SEQUENCE{
    memTag              MemTag,                 -- Memory Tag ( 8 Bytes )
    fill                BITSTRING(SIZE(5)),
    permission          Permission,             -- Protection mode
    maxMemorySize       INTEGER(0...4294967295),-- Memory allocation size
    initialValue        OCTET STRING            -- Initial set value
```

```
}

MemoryAllocInfoWithCredence ::= SEQUENCE{
    memTag              MemTag,                     -- Memory Tag ( 8 Bytes )
    fill                BITSTRING(SIZE(5)),
    permission          Permission,                 -- Protection mode
    maxMemorySize       INTEGER(0...4294967295), -- Memory allocation size
    accessCredential    OCTET STRING(SIZE(8)), -- Password
    initialValue        OCTET STRING            -- Initial set value
}

MemDataListWithCredence ::=SEQUENCE OF MemDataWithCredence

MemTagListWithCredence ::=SEQUENCE OF MemTagWithCredence

MemDataWithCredence ::= SEQUENCE{
    memTagWithCredence      MemTagWithCredence,
    data                    OCTET STRING
}

MemTagWithCredence::= SEQUENCE{
    memTag                  MemTag,
    accessCredential        OCTET STRING(SIZE(8))
}

MemTagList::= SEQUENCE OF MemTag.

MemDataList ::= SEQUENCE OF MemData.

MemData::=SEQUENCE{
    memTag              MemTag,
    data                OCTET STRING
}

MemTag::= OCTET STRING(SIZE(8))
```

3.2.4 Relationship with Other Standards

The relationships with other DSRC related standards used this application are shown below.

Table 3.2-43 Relationship with other DSRC related standards

	DSRC related standard	Content using in this application
1	DSRC standard	AID=18 (DSRC Application Sub Layer)
2	NCP of ASL	LPCP (Local Port Control Protocol)
3	port number of LPCP	0x0C18
4	transaction service	Request-response type transaction service in two transaction service provided by LPCP. When the message size is larger than MTU size of LPCP, message segmentation/re-assembly function of LPP is used.

Note : response command corresponding to request command is identify by request-response type transaction

3.2.5 Communication procedures

This subsection describes communication procedures for accessing the memory in the OBE using the OBE memory access application.

3.2.5.1 Acquiring memory information

(1) The roadside system sends to the OBE the memory resource information acquisition request command (resourceInfoRequest) containing the memory tag list used later.

(2) When memory tags specified in the received memory resource information acquisition request command are already registered (that is, their data storage memory areas are already allocated) in the OBE, the OBE stores the attributes and sends them to the roadside system using the memory resource information acquisition response command (resourceInfoResponse).

(3) The OBE memory resource information acquisition response command sends the maximum data size receivable in the OBE, remaining number of roadside system allocatable memory tags, remaining capacity of the roadside system allocatable memory area and upper limit number of batch memory tags. In the OBE not incorporating the OBE

memory allocation function, however, “0” is set to the remaining number of roadside system allocatable memory tags and remaining capacity of the roadside system allocatable memory area.

(4) When every memory tag specified by the memory resource information acquisition request command does not exist in the step (2), the OBE sets “0” to the number of memory tags in the memory resource information acquisition response command sent to the roadside system.

3.2.5.2 Allocating roadside system allocatable memory

(1) The roadside system sends to the OBE the OBE memory allocation request command (memoryAllocRequest) for allocating the roadside system allocatable memory area.

(2) The OBE allocates data storage memory areas based on memory tags specified in the received memory allocation request command, and sets the attributes. When the attributes are set normally, the OBE sends the memory allocation response command (memoryAllocResponse) to the roadside system.

(3) When the OBE does not incorporate the memory allocation function in the step (2), the OBE sends to the roadside system the OBE denial response command (obuDenialResponse) notifying the status 12 “Unsupported command”.

(4) When the memory tag specified in the memory allocation request command is an OBE controlling memory tag in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(5) When the memory tag specified in the memory allocation request command is already registered (that is, its data storage memory area is already allocated) in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(6) When memory allocation is disabled due to insufficient capacity of the roadside system allocatable memory in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 5 “Insufficient free space in OBE”.

(7) When the OBE cannot handle the memory allocation size specified by the roadside system in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 9 “Access control violation”.

(8) When memory allocation is not executed normally in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 3 “OBE memory allocation/free failed”.

3.2.5.3 Making free roadside system allocatable memory

(1) The roadside system sends the memory free request command (memoryFreeRequest) to the OBE.

(2) The OBE makes data storage memory areas free based on memory tags specified in the received memory free request command. When data storage memory areas are made free normally, the OBE sends the memory free response command (memoryFreeResponse) command to the roadside system.

(3) When the OBE does not incorporate the memory free function in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 12 “Unsupported command”.

(4) When a memory tag whose protection mode is “SPF essential” is specified in the received memory free request command without using the SPF in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 11 “SPF violation”.

(5) When the memory tag specified in the memory free request command is not registered (that is, its data storage memory area is not allocated) in the OBE in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(6) When the password is set in the memory tag specified in the memory free request command in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 10 “Password disagreed”.

(7) When memory free is not executed normally in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 3 “OBE memory allocation/free failed”.

3.2.5.4 Reading data from one data storage memory area

(1) The roadside system sends the memory read request command (readRequest) to the OBE.

(2) The OBE reads one data storage memory area specified in the received memory read request command, and sends the memory read response command (readResponse) to the roadside system.

(3) When a memory tag whose protection mode is “SPF essential” is specified in the received memory read request command without using the SPF in the step (2), the OBE sends to the roadside system the OBE denial response command (obuDenialResponse) notifying the status 11 “SPF violation”.

(4) When the memory tag specified in the memory read request command is not registered (that is, its data storage memory area is not allocated) in the OBE in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(5) When the password is set in the memory tag specified in the memory read request command in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 10 “Password disagreed”.

(6) When a memory tag whose protection mode is “write only” is specified in the memory read request command in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 8 “Protection mode violation”.

(7) When reading from one data storage memory area in the OBE has failed in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 1 “Read from OBE memory failed”.

(8) It is recommended in the step (1) to acquire in advance the attributes of the memory tag (data storage memory area) to be read using the memory resource information acquisition request command (resourceInfoRequest), etc., and then confirm that the size to be read is allowed in both the roadside system and the OBE.

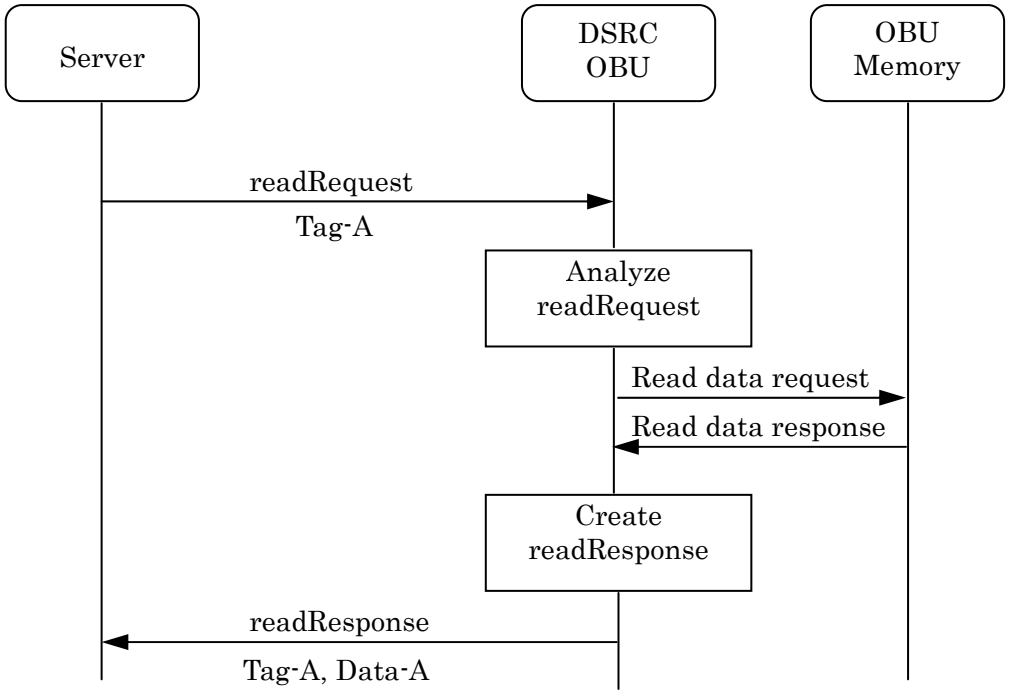


Figure3.2-1 Example sequence of reading data from one data storage memory area

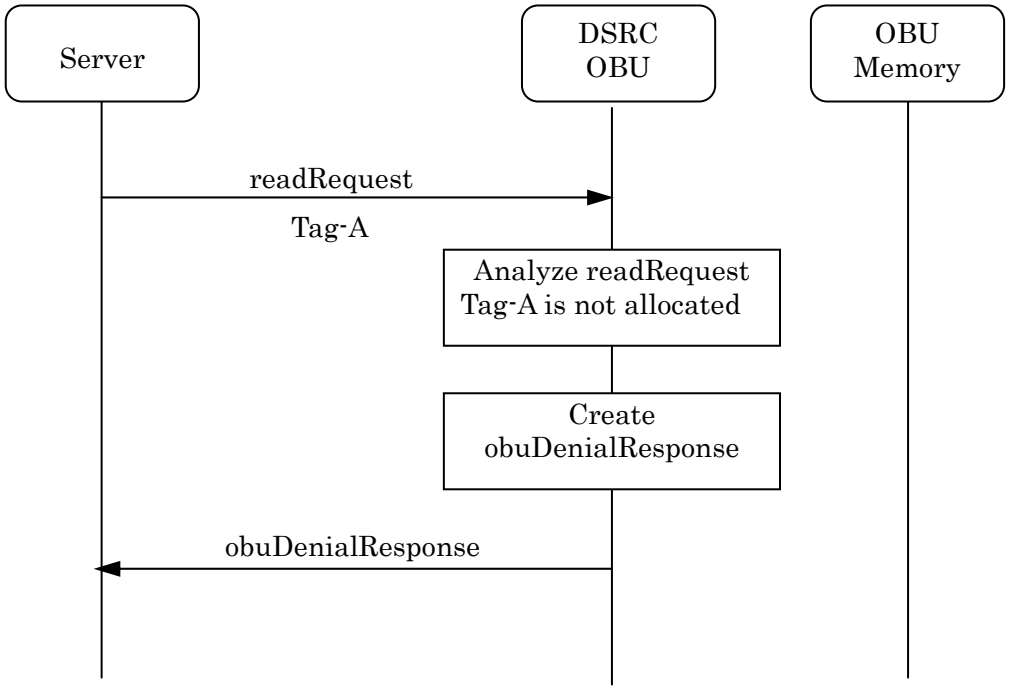


Figure 3.2-2 Example sequence of OBU denial response

3.2.5.5 Writing data to one data storage memory area

(1) The roadside system sends the memory write request command (writeRequest) to the OBE.

(2) The OBE writes data to one data storage memory area specified in the received memory write request command, and sends the memory write response command to the roadside system. Existing data is overwritten with the received data.

(3) When a memory tag whose protection mode is “SPF essential” is specified in the received memory write request command without using the SPF in the step (2), the OBE sends to the roadside system the OBE denial response command (obuDenialResponse) notifying the status 11 “SPF violation”.

(4) When the memory tag specified in the memory write request command is not registered (that is, its data storage memory area is not allocated) in the OBE in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(5) When the password is set in the memory tag specified in the memory write request command in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 10 “Password disagreed”.

(6) When a memory tag whose protection mode is “read only” is specified in the memory write request command in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 8 “Protection mode violation”.

(7) When data larger than the data storage area is specified in the memory write request command in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 9 “Access control violation”.

(8) When writing to one data storage memory area in the OBE has failed in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 2 “Write to OBE memory failed”.

(9) It is recommended in the step (1) to acquire in advance the attributes of the memory tag (data storage memory area) to be written using the memory resource information acquisition request command (resourceInfoRequest), etc., and then confirm that the size to be written is allowed in both the roadside system and the OBE.

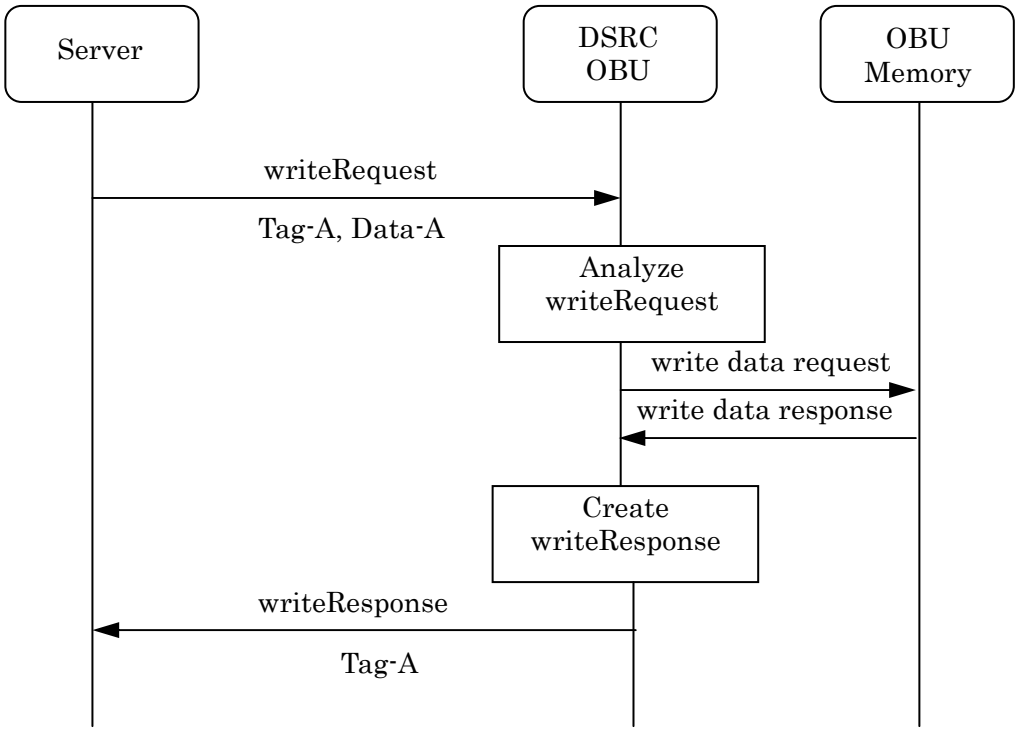


Figure 3.2-3 Example sequence of Example sequence of writing data to one data storage memory area

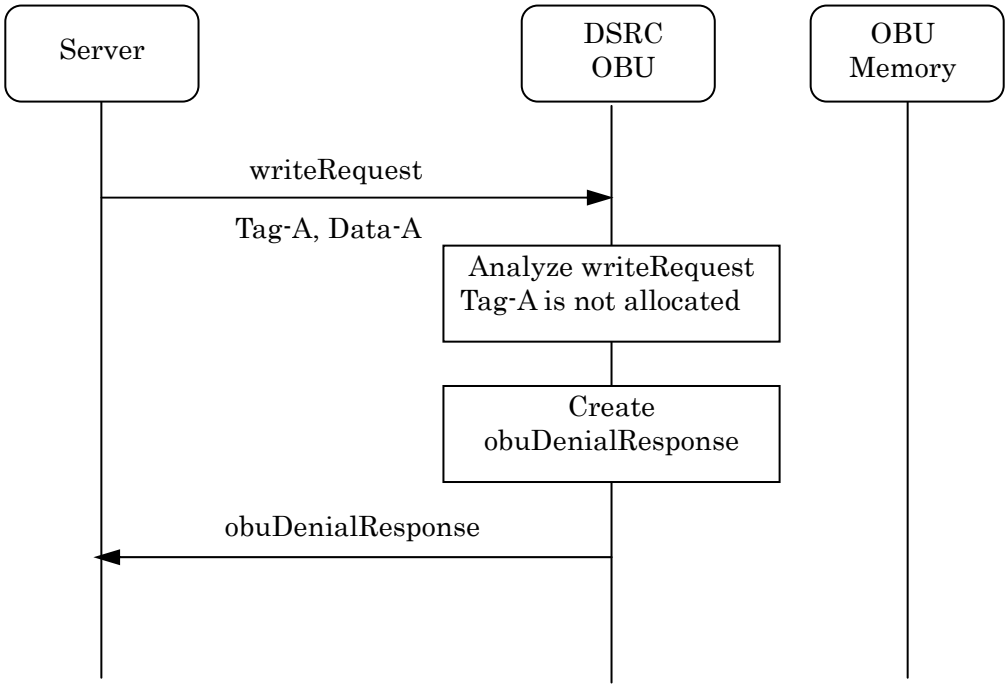


Figure3.2-4 Example sequence of OBU denial response

3.2.5.6 Reading data simultaneously from one or more data storage memory areas

(1) The roadside system sends the bulk memory read request command (readBulkRequest) to the OBE.

(2) The OBE reads data storage memory areas specified in the received bulk memory read request command, and sends the bulk memory read response command (readBulkResponse) to the roadside system for notifying memory tags and stored data read successfully.

(3) When data cannot be read from a memory tag due to no corresponding memory tag, protection mode violation, password disagreed, read data size error or any another reason in the step (2), the OBE aborts processing of the corresponding memory tag, and continues processing of the next memory tag.

(4) When the number of memory tags specified in the bulk memory read request command exceeds the maximum number of tags handled simultaneously by the OBE in the step (2), the OBE sends to the roadside system the OBE denial response command (obuDenialResponse) notifying the status 7 “Maximum available number of tags over in batch processing”.

(5) When every requested memory tag does not exist in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(6) When reading from every requested memory tag has failed in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 1 “Read from OBE memory failed”.

(7) It is recommended in the step (1) to acquire in advance the attributes of the memory tags (data storage memory areas) to be read using the memory resource information acquisition request command (resourceInfoRequest), etc., and then confirm that the size to be read is allowed in both the roadside system and the OBE.

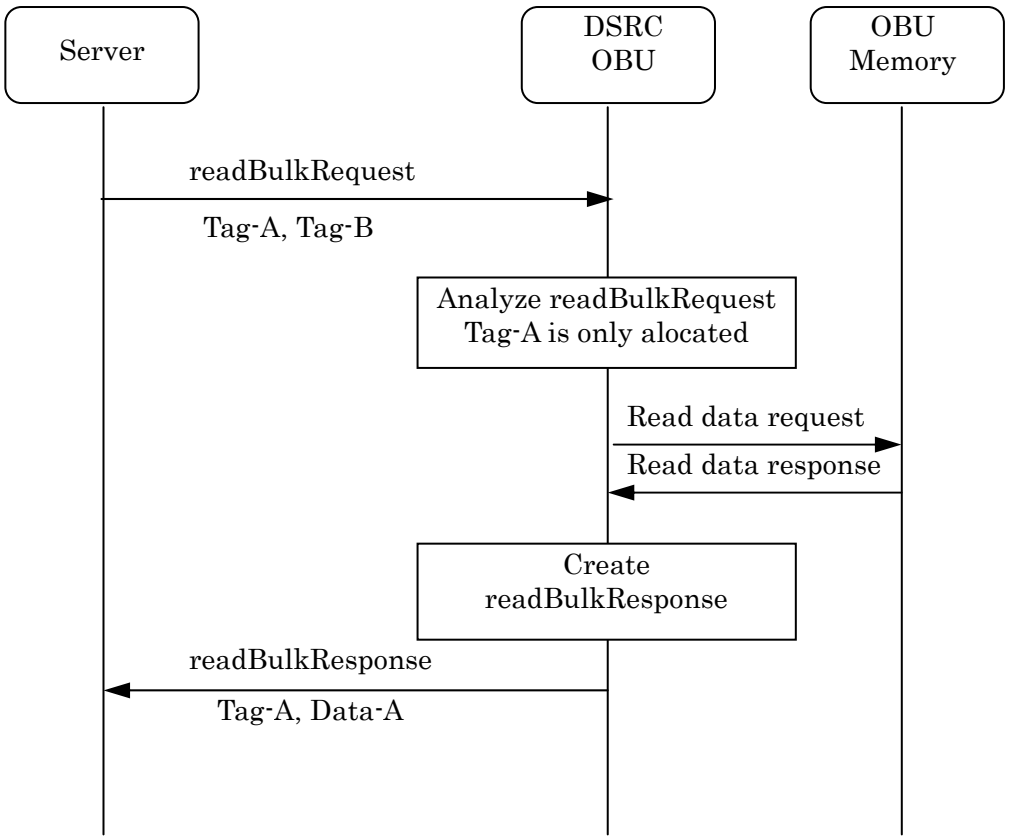


Figure 3.2-5 Example sequence of reading data simultaneously from one or more data storage memory areas

3.2.5.7 Writing data simultaneously to one or more data storage memory areas

(1) The roadside system sends the bulk memory write request command (writeBulkRequest) to the OBE.

(2) The OBE writes data storage memory areas specified in the received bulk memory write request command, and sends the bulk memory write response command (writeBulkResponse) to the roadside system. for notifying memory tags written successfully. Existing data are overwritten with the received data.

(3) When data cannot be written to a memory area due to no corresponding memory tag, protection mode violation, password disagreed, write data size error or any another reason in the step (2), the OBE aborts processing of the corresponding memory tag, and continues processing of the next memory tag.

(4) When the number of memory tags specified in the bulk memory write request command exceeds the maximum number of tags handled simultaneously by the OBE in the step (2), the OBE sends to the roadside system the OBU denial response command (obuDenialResponse) notifying the status 7 “Maximum available number of tags over in batch processing”.

(5) When every requested memory tag does not exist in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(6) When writing to every requested memory tag has failed in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 1 “Write to OBE memory failed”.

(7) It is recommended in the step (1) to acquire in advance the attributes of the memory tags (data storage memory areas) by using the memory resource information acquisition request command (resourceInfoRequest), and then confirm that the size to be written is allowed in both the roadside system and the OBE.

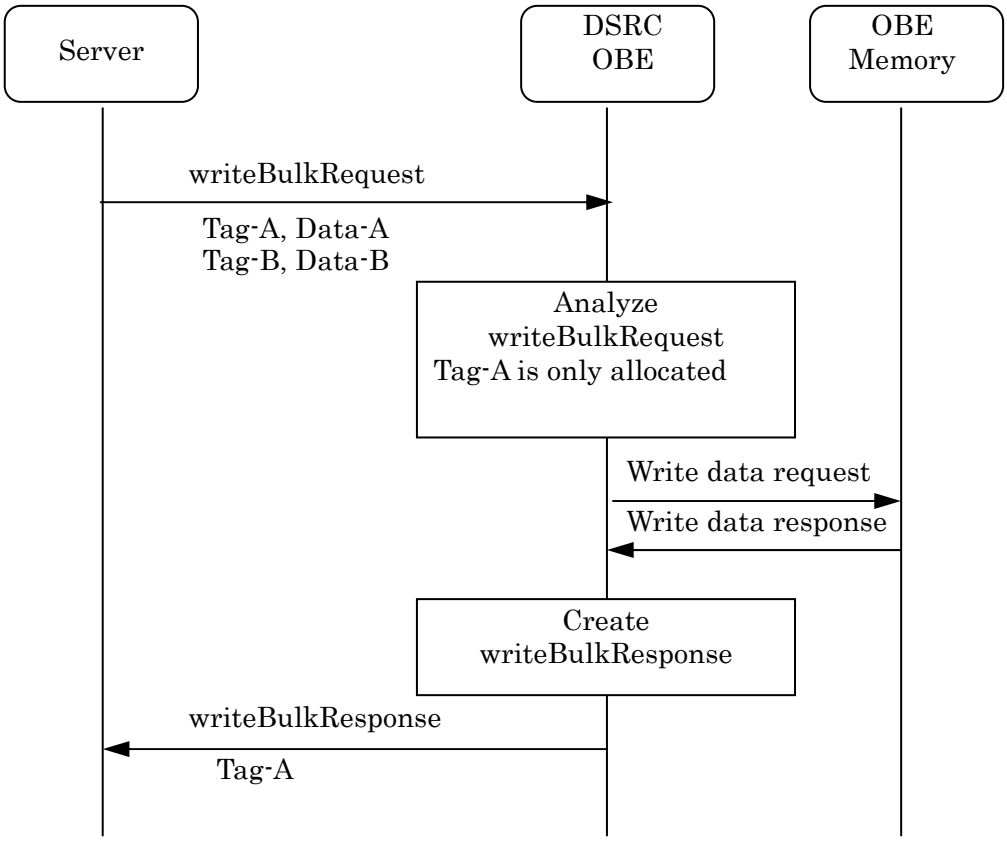


Figure 3.2-6 Example sequence of writing data simultaneously to one or more data storage memory areas

3.2.5.8 Allocating roadside system allocatable memory with password

(1) The roadside system sends to the OBE the OBE memory allocation with password request command (memoryAllocRequestWithCredence) for allocating the roadside system allocatable memory area with password.

(2) The OBE allocates data storage memory areas based on memory tags specified in the received memory allocation with password request command, and sets the attributes (include password). When the attributes are set normally, the OBE sends the memory allocation with password response command (memoryAllocResponseWithCredence) to the roadside system.

(3) When the OBE does not incorporate the memory allocation function in the step (2), the OBE sends to the roadside system the OBE denial response command (obuDenialResponse) notifying the status 12 “Unsupported command”.

(4) When the memory tag specified in the memory allocation with password request command is an OBE controlling memory tag in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(5) When the memory tag specified in the memory allocation with password request command is already registered (that is, its data storage memory area is already allocated) in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(6) When memory allocation is disabled due to insufficient capacity of the roadside system allocatable memory in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 5 “Insufficient free space in OBE”.

(7) When the OBE cannot handle the memory allocation size specified by the roadside system in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 9 “Access control violation”.

(8) When memory allocation is not executed normally in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 3 “OBE memory allocation/free failed”.

3.2.5.9 Making free roadside system allocatable memory with password

(1) The roadside system sends the memory free with password request command (memoryFreeRequestWithCredence) to the OBE.

(2) The OBE makes data storage memory areas free based on memory tags specified in the received memory free with password request command. When data storage memory area whose password is same to received one are made free normally, the OBE sends the memory free with password response command (memoryFreeResponse- WithPassword) command to the roadside system.

(3) When the OBE does not incorporate the memory free with password function in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 12 “Unsupported command”.

(4) When a memory tag whose protection mode is “SPF essential” is specified in the received memory free request command without using the SPF in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 11 “SPF violation”.

(5) When the memory tag specified in the memory free with password request command is not registered (that is, its data storage memory area is not allocated) in the OBE in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(6) When the password is not set in the memory tag specified in the memory free with password request command in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 10 “Password disagreed”.

(7) When the password of the memory tag specified in the memory free with password request command is different from received one in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 10 “Password disagreed”.

(8) When memory free is not executed normally in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 3 “OBE memory allocation/free failed”.

3.2.5.10 Reading data from one data storage memory area with password

(1) The roadside system sends the memory read with password request command (readRequestWithCredence) to the OBE.

(2) The OBE reads one data storage memory area specified in the received request command,

and when password of stored data is same to received one, the OBU sends the memory read with password response command (readResponseWithCredence) to the roadside system.

(3) When the OBE does not incorporate the password function in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 12 “Unsupported command”.

(4) (4) When the memory tag specified in the request command is not registered (that is, its data storage memory area is not allocated) in the OBE in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(5) When a memory tag whose protection mode is “SPF essential” is specified in the received memory read request command without using the SPF in the step (2), the OBE sends to the roadside system the OBE denial response command (obuDenialResponse) notifying the status 11 “SPF violation”.

(6) When the password is not set in the memory tag specified in the request command in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 10 “Password disagreed”.

(7) When the password in the memory tag specified in the request command is different from received one in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 10 “Password disagreed”.

(8) When a memory tag whose protection mode is “write only” is specified in the request command in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 8 “Protection mode violation”.

(9) When reading from one data storage memory area in the OBE has failed in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 1 “Read from OBE memory failed”.

(10) It is recommended in the step (1) to acquire in advance the attributes of the memory tag (data storage memory area) to be read using the memory resource information acquisition request command (resourceInfoRequest), etc., and then confirm that the size to be read is allowed in both the roadside system and the OBE.

3.2.5.11 Writing data to one data storage memory area with password

(1) The roadside system sends the memory write with password request command (writeRequestWithCredence) to the OBE.

(2) The OBE writes data to one data storage memory area specified in the received request

command when the password of storage memory area is same to received one, and sends the memory write with password response command to the roadside system. Existing data is overwritten with the received data.

(3) When the OBE does not incorporate the password function in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 12 “Unsupported command”.

(4) When the memory tag specified in the request command is not registered (that is, its data storage memory area is not allocated) in the OBE in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(5) When a memory tag whose protection mode is “SPF essential” is specified in the received request command without using the SPF in the step (2), the OBE sends to the roadside system the OBE denial response command (obuDenialResponse) notifying the status 11 “SPF violation”.

(6) When the password is not set in the memory tag specified in the request command in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 10 “Password disagreed”.

(7) When the password in the memory tag specified in the request command is different from received one in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 10 “Password disagreed”.

(8) When a memory tag whose protection mode is “read only” is specified in the request command in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 8 “Protection mode violation”.

(9) When data larger than the data storage area is specified in the request command in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 9 “Access control violation”.

(10) When writing to one data storage memory area in the OBE has failed in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 2 “Write to OBE memory failed”.

(11) It is recommended in the step (1) to acquire in advance the attributes of the memory tag (data storage memory area) to be written using the memory resource information acquisition request command (resourceInfoRequest), etc., and then confirm that the size to be written is allowed in both the roadside system and the OBE.

3.2.5.12 Reading data simultaneously from one or more data storage memory area with password

(1) The roadside system sends the bulk memory read with password request command (readBulkRequestWithCredence) to the OBE.

(2) The OBE reads data storage memory areas whose password is same specified in the received bulk memory read request command, and sends the bulk memory read with password response command (readBulkResponseWithCredence) to the roadside system for notifying memory tags and stored data read successfully.

(3) When data cannot be read from a memory tag due to no corresponding memory tag, protection mode violation, password disagreed, read data size error or any another reason in the step (2), the OBE aborts processing of the corresponding memory tag, and continues processing of the next memory tag.

(4) When the OBE does not incorporate the password function in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 12 “Unsupported command”.

(5) When the number of memory tags specified in the request command exceeds the maximum number of tags handled simultaneously by the OBE in the step (2), the OBE sends to the roadside system the OBE denial response command (obuDenialResponse) notifying the status 7 “Maximum available number of tags over in batch processing”.

(6) When every requested memory tag does not exist in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(7) When reading from every requested memory tag has failed in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 1 “Read from OBE memory failed”.

(8) It is recommended in the step (1) to acquire in advance the attributes of the memory tags (data storage memory areas) to be read using the memory resource information acquisition request command (resourceInfoRequest), etc., and then confirm that the size to be read is allowed in both the roadside system and the OBE.

3.2.5.13 Writing data simultaneously to one or more data storage memory areas with password

(1) The roadside system sends the bulk memory write with password request command (writeBulkRequestWithCredence) to the OBE.

(2) The OBE writes data storage memory areas specified in the received request command when the password of each storage memory area is same to received one, and sends the bulk memory write with password response command (writeBulkResponseCredence) to the roadside system. for notifying memory tags written successfully. Existing data are overwritten with the received data.

(3) When data cannot be written to a memory area due to no corresponding memory tag, protection mode violation, password disagreed, write data size error or any another reason in the step (2), the OBE aborts processing of the corresponding memory tag, and continues processing of the next memory tag.

(4) When the OBE does not incorporate the password function in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 12 “Unsupported command”.

(5) When the number of memory tags specified in the request command exceeds the maximum number of tags handled simultaneously by the OBE in the step (2), the OBE sends to the roadside system the OBU denial response command (obuDenialResponse) notifying the status 7 “Maximum available number of tags over in batch processing”.

(6) When every requested memory tag does not exist in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 6 “No corresponding memory tag”.

(7) When writing to every requested memory tag has failed in the step (2), the OBE sends to the roadside system the OBE denial response command notifying the status 1 “Write to OBE memory failed”.

(8) It is recommended in the step (1) to acquire in advance the attributes of the memory tags (data storage memory areas) by using the memory resource information acquisition request command (resourceInfoRequest), and then confirm that the size to be written is allowed in both the roadside system and the OBE.

3.3 IC Card Access Application

3.3.1 Function Overview

The IC card access application provides functions to access the IC card using methods specified in ISO/IEC7816 in accordance with requests from the roadside system. The IC card access application handles only IC card in conformity to ISO/IEC7816.

The IC card access application concretely provides the following three functions:

(1) Application start function to perform the IC card initialization processing, etc.

The OBE activates the IC card based on the initialization request received from the roadside system. The OBE edits the ATR (reset response) received from the IC card into a message, and transfers it to the roadside system.

(2) IC card command send/receive function to send and receive ISO/IEC7816 commands to/from IC card.

The OBE transfers the CommandAPDU received from the roadside system to the IC card based on the EMV level 1 specification. The OBE edits the ResponseAPDU received from the IC card into a message, and transfers it to the roadside system.

(3) Application termination function to perform the IC card inactivation termination processing, etc.

The OBE inactivates the IC card based on the termination request received from the roadside system.

3.3.2 Command

3.3.2.1 Command System

Commands (ICCAccessCommand) of the IC card access application consist of normal commands, OBE denial response command from the OBE, and committed information acquisition commands. The normal commands consist of " application start request command, IC card command send command, application termination request command, application start response command, IC card response send command and application termination response command". Committed information acquisition commands consist of the committed information acquisition request command and committed information acquisition response command. The version number is "1" for the IC card access application having the specification described in this document.

3.3.2.2 Command Format

3.3.2.2.1 Normal Command

3.3.2.2.1.1 Application Start Request Command

This command is used when the roadside system notifies the OBE that the IC card access application processing is started. Table 3.3-1 shows the command format.

Table 3.3-1 Application Start Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type initRequest(3)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody "opCommandBody" (0)							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier "operationCommand(1)" to indicate the normal

command.

(3) Operation Type

This field is set to an identifier “initRequest(3)” to indicate the request command of application start.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

This field indicates the data length (0) of succeeding operation command body. The unit is octet.

3.3.2.2.1.2 Application Start Response Command

This command is used when the OBE notifies the roadside system of response to application start request and IC card initialization result. Table 3.3-2 shows the command format.

Table 3.3-2 Application Start Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type initResponse(131)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody value of ATR obtained from IC card							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “initResponse(131)” to indicate the response command

of application start.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) The data length of OpCommandBody

This field indicates the data length of succeeding operation command body. The unit is octet. In this command, this size is extended based on ASN.1 coding rules.

(b) The content of OpCommandBody

This field is set to the value of ATR (reset response) obtained from IC card.

3.3.2.2.1.3 IC Card Command Send Command

This command is used when the roadside system sends IC card data to the OBE. Table 3.3-3 shows the command format.

Table 3.3-3 IC Card Command Send Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type iCCCommand(0)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody CommandAPDU of ISO/IEC7816-4 passed to IC card							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “iCCCommand(0)” to indicate the send command of IC

card command.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) The data length of OpCommandBody

This field indicates the data length of succeeding operation command body. The unit is octet. In this command, this size is extended based on ASN.1 coding rules

(b) The content of OpCommandBody

This field is set to the CommandAPDU of ISO/IEC7816-4 passed to IC card.

3.3.2.2.1.4 IC Card Response Send Command

This command is used when the OBE sends data received from the IC card to the roadside system. Table 3.3-4 shows the command format.

Table 3.3-4 IC Card Response Send Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type iCCResponse(128)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody							
:	The content of OpCommandBody ResponseAPDU of ISO/IEC7816-4 obtained from IC card							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “iCCResponse(128)” to indicate the response send

command of IC card.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) The data length of OpCommandBody

This field indicates the data length of succeeding operation command body. The unit is octet. In this command, this size is extended based on ASN.1 coding rules.

(b) The content of OpCommandBody

This field is set to the ResponseAPDU of ISO/IEC7816-4 obtained from IC card.

3.3.2.2.1.5 Application Termination Request Command

This command is used when the roadside system notifies the OBE that the IC card access application processing is terminated. Table 3.3-5 shows the command format.

Table 3.3-5 Application Termination Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type endRequest(2)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody (0)							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “endRequest(2)” to indicate the request command of application termination.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

This field indicates the data length (0) of succeeding operation command body. The unit is octet.

3.3.2.2.1.6 Application Termination Response Command

This command is used when the OBE notifies the roadside system of response to application termination request. Table 3.3-6 shows the command format.

Table 3.3-6 Application Termination Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	operation Type endResponse(130)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody (0)							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “endResponse(3)” to indicate the response command of Application termination.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

This field indicates the data length (0) of succeeding operation command body. The unit is octet.

3.3.2.2.2 OBE Denial Response Command

The OBE denial response command is used when the OBE notifies the roadside system of its abnormal status. Table 3.3-7 shows the command format.

Table 3.3-7 OBE Denial Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type obuDenialResponse(255)							
3	Status							
4	The length of supplement information							
5 :	The contents of supplement information							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “obuDenialResponse(255)” to indicate the OBE denial response.

(3) Status

This field is set to the reason of denial response. For details, refer to Table 3.3-8.

Table 3.3-8 Status Code

Status code	Description
0-1	For future use
2	ICC insertion error (reverse side, reverse direction, etc.)
3	ICC no response (access error or retry over)
4	The version incompatible
5	ICC not inserted
6	ICC data error (wrong IC Card)

7-15	For future use
16	Illegal command
17-254	For future use
255	Other OBE internal error

(4) Supplement Information

(a) The length of supplement information

This field is set to the length of successing supplement information. The unit is octet. When there are no supplement information (this case is default case), this filed is set to value “0”.

(b) The contents of supplement information

This field is set to free information (the maximum length is 127 octets) as supplement information. When version number is not same, this field is set to own version (“versionIndex” parameter).

3.3.2.2.3 Committed Information Acquisition Commands

3.3.2.2.3.1 Committed Information Acquisition Request Command

The committed information acquisition request command is issued from the roadside system to the OBE to acquire the committed information stored in the OBE. Table 3.3-9 shows the command format.

Table 3.3-9 Committed Information Acquisition Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill			
2	Command Type accreditationInfoCommand(2)							
3	Operation Type accreditationInfoRequest(0)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody (0)							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “accreditationInfoCommand(2)” to indicate the committed information acquisition request.

(3) Operation Type

This field is set to an identifier “accreditationInforRequest(0)” to indicate the request command of committed information acquisition.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) The data length of OpCommandBody

This field indicates the data length (0) of succeeding operation command body. The unit is octet.

3.3.2.2.3.2 Committed Information Acquisition Response Command

The committed information acquisition response command is issued from the OBE to the roadside system as response to the committed information acquisition request command for notifying the roadside system of the committed information stored in the OBE. Table 3.3-10 shows the command format.

Table 3.3-10 Committed Information Acquisition Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill			
2	Command Type accreditationInfoCommand(2)							
3	Operation Type accreditationInfoResponse(128)							
4	Security Profile plainText(0)							
5	The data length of OpCommandBody (1)							
6	The content of OpCommandBody OBE committed information “OBUAccreditationInfo” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “accreditationInfoCommand(2)” to indicate the committed information acquisition operation.

(3) Operation Type

This field is set to an identifier “accreditationInfoResponse(129)” to indicate the response command of the committed information acquisition.

(4) Security Profile

This field is set to plainText(0) as attribute of operation command body in this command.

(5) Operation Command Body

(a) The data length of OpCommandBody

This field indicates the data length (1) of succeeding operation command body. The unit is octet.

(b) The content of OpCommandBody

This field is set to the OBE committed information (“OBUAccreditationInfo” format).

3.3.3 Definition of Parameter Types

```
ICCAccessCommand::=SEQUENCE {
    versionIndex                Version,
    accessCommand               AccessCommand
}

Version::=SEQUENCE{
    version     INTEGER(0..15),         -- set up 1 initially
    fill        BIT STRING(SIZE(4))     -- value of encoding is set up 0
}

AccessCommand::=CHOICE{
    dummy                       [0]     NULL,   -- Not use
    operationCommand            [1]     OperationCommand,
    accreditationInfoCommand    [2]     AccreditationInfoCommand,
    dummy                       [3-254] NULL,   -- For future use
    obuDenialResponse           [255]   ObuDenialResponse
}

OperationCommand::=SEQUENCE{
    opCommandType               OpCommandType,
    opSecurityProfile           OpSecurityProfile,
    opCommandBody               OCTET STRING
}

OpCommandType::=ENUMERATED{
    iCCCommand                  (0),    -- IC Card Command Sent
    reservedFor future use      (1),    -- For future use
    endRequest          (2),            -- Application End Request
    initRequest         (3),            -- Application Start Request
    reservedFor future use      (4-127), -- For future use
    iCCResponse                 (128),  -- IC Card Response Send
    reservedFor future use      (129),  -- For future use
```

```
    endResponse                 (130),    -- Application End Response
    initResponse                (131),    -- Application Start Response
    reservedFor future use      (132-255)       -- For future use
}

ObuDenialResponse::=SEQUENCE{
    status              INTEGER(0...255),               -- Status Code
    supplementInfo      OCTET STRING(SIZE(0...255))-- supplement information
}

AccreditationInfoCommand::=SEQUENCE{
    acCommandType               AcCommandType,
    acSecurityProfile           OpSecurityProfile,
    acCommandBody       OCTET STRING
}

AcCommandType::=ENUMERATED{
    accreditationInfoRequest    (0),
                -- Accreditation Information Acquisition Request
    reservedFor future use      (1-127),        -- For future use
    accreditationInfoResponse   (128),
                -- Accreditation Information Acquisition Response
    reservedFor future use      (129 -255)      -- For future use
}

OBUAccreditationInfo::=SEQUENCE{
    emvIcc      BOOLEAN,
                        -- presence or absence of accreditation information of EMV
    fill        BIT STRING(SIZE(7))     -- For future use
}
```

3.3.4 Relationship with Other Standard

The relationships with other DSRC related standards used this application are shown below.

Table 3.3-11 Relationship with other DSRC related standards

	DSRC related standard	Content using in this application
1	DSRC Standard	AID=18 (DSRC Application Sub Layer)
2	NCP of ASL	LPCP (Local Port Control Protocol)
3	Port number of LPCP	Contact IC; 0x0C10 Contactless IC; 0x0C11
4	Transaction Service	The relationships of the other DSRC related standards using this application are shown in below.

3.3.5 Communication Procedures

This subsection describes the communication procedures of the IC card access application.

Note: Annex.C-5 shows an example of settlement processing combining the basic sequences (1) to (3).

(1) IC card initialization processing

(a) The roadside system sends the application start request command to the OBE.

(b) When receiving the application start request command, the OBE executes the IC card activation processing, and notifies the roadside system using the application start response command of the ATR value acquired from the IC card.

(c) When IC card activation has failed (because the IC card is not inserted, the IC card is inserted in the reverse direction, etc.) in the step (b), the OBE sends the OBE denial response command instead of the application start response command to the roadside system.

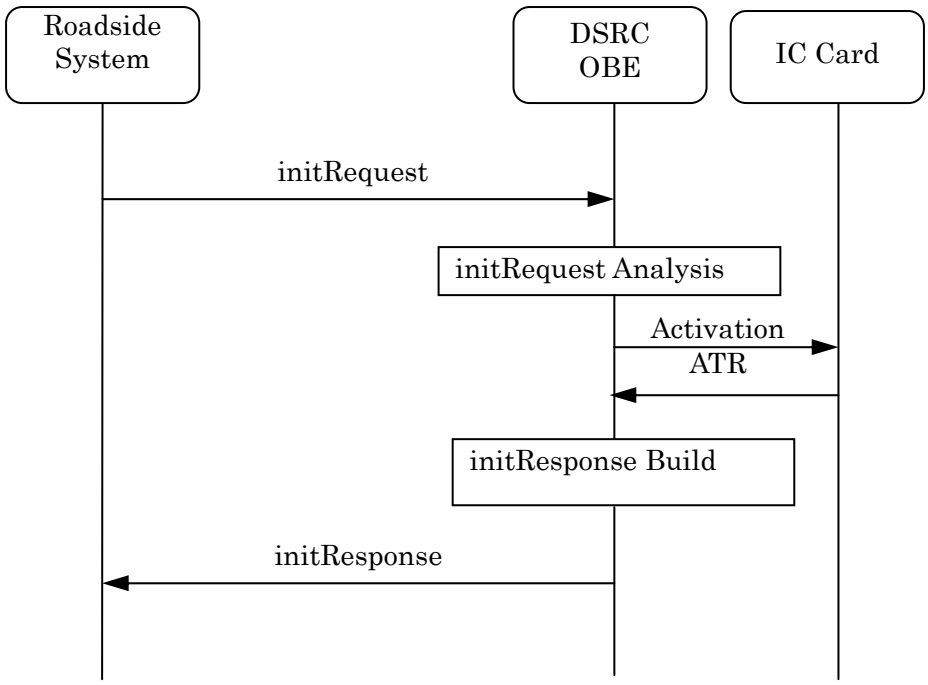


Figure 3.3-1 Sequence example of initiation process of IC Card

(2) IC card data processing (APDU command processing)

(a) The roadside system notifies the OBE of "CommandAPDU" using the IC card command send command.

(b) When receiving the IC card command send command, the OBE acquires

"CommandAPDU", transfers it to the IC card based on the EMV specification, and waits for response from the IC card.

(c) When receiving "ResponseAPDU" from the IC card, the OBE stores it in the IC card response send command, and notifies the roadside system of "ResponseAPDU".

(d) When command transfer with the IC card is not executed normally in the step (b) or (c), the OBE notifies the OBE denial response command instead of the IC card response send command to the roadside system. After that, the OBE inactivates the IC card, and waits for an application start request from the roadside system.

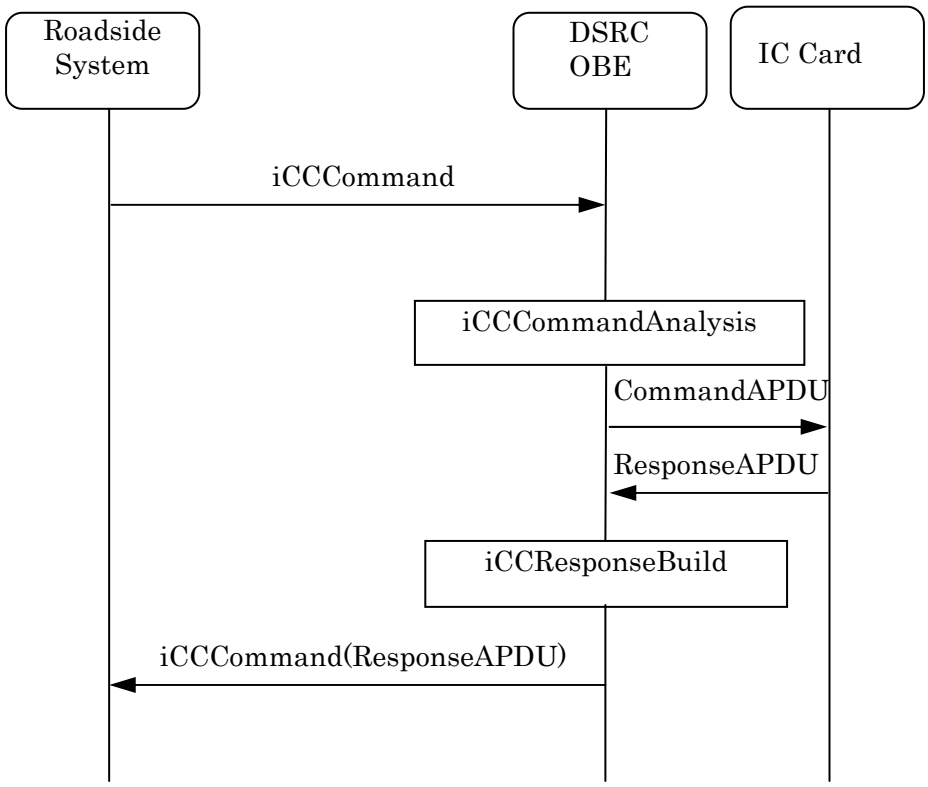


Figure 3.3-2 Sequence example of IC card data process

(3) IC card termination processing

(a) The roadside system sends the application termination request command to the OBE.

(b) When receiving the application termination request command, the OBE executes the IC card inactivation processing. When the inactivation processing is finished, the OBE notifies the roadside system of IC card inactivation using the application termination response command.

(c) When IC card inactivation has failed in the step (b), the OBE sends the OBE denial response command instead of the application termination response command to the

roadside system.

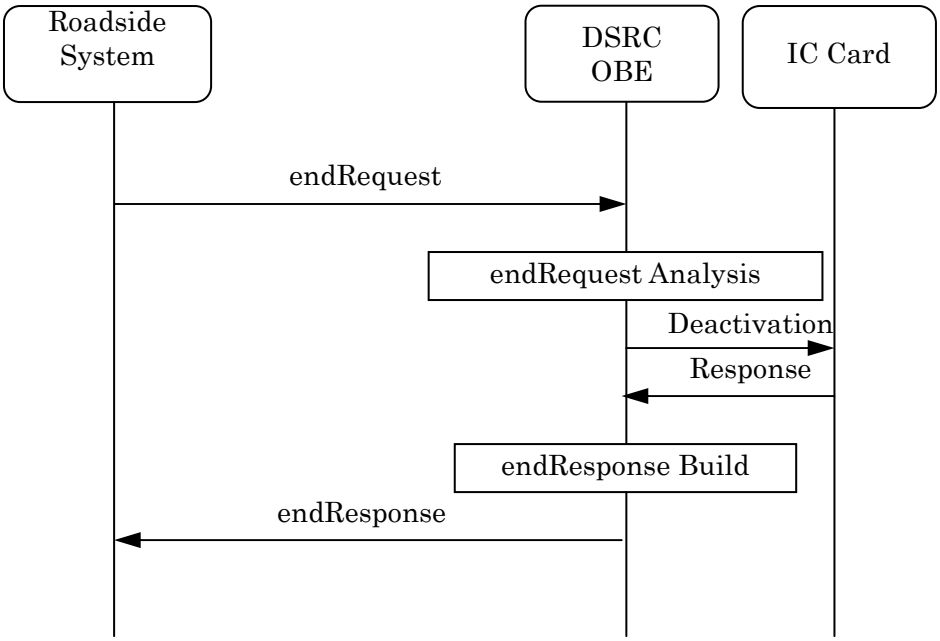


Figure 3.3-3 Sequence example of End process of IC card

(4) OBE committed information acquisition processing

(a) For acquiring the committed information, the roadside system sends the committed information acquisition request command to the OBE before the IC card initialization processing is started.

(b) When receiving the committed information acquisition request command, the OBE checks the stored committed information, and notifies the roadside system of the committed information using the committed information acquisition response command.

(c) When receiving the committed information acquisition response command, the roadside system checks the stored committed information, and executes the IC card initialization processing if the OBE holds required committed information.

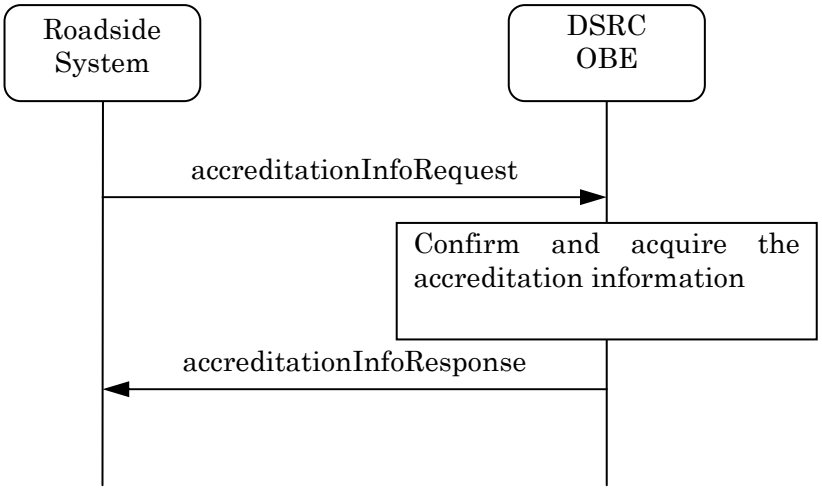


Figure 3.3-4 Sequence example of Acquisition process of accreditation information

3.4 Push-type Information Delivery Application

3.4.1 Outline of functions

The push-type information delivery application sends a content or content position from the server in the roadside system to the client in the OBE, and automatically executes processing in accordance with the received content type in the client.

The method to distribute a content itself is called “content push”, and the method to distribute a content position (such as URL) and acquire a content separately using the HTTP, etc. is called “pseudo push”.

This subsection describes the content push and pseudo push procedures.

(1) Realization example of the content push

A realization example of the content push is as follows

(a) After the DSRC (dedicated short range communication) route is established,

(b) the push server transmits a content, and

(c) the push client analyzes the content type and starts up the corresponding application.

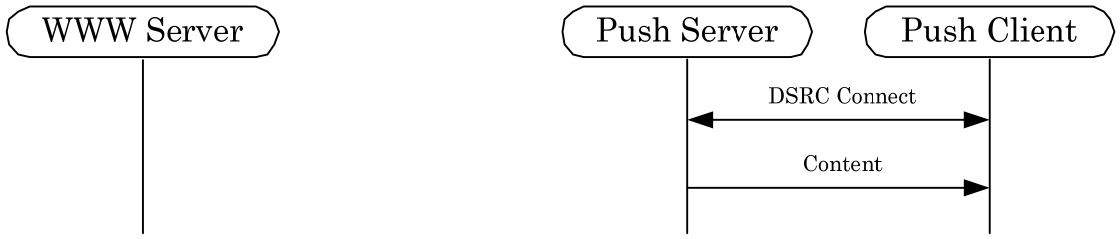


Figure 3.4-1 example sequence of “content push”

(2) Realization example of the pseudo push

A realization example of the pseudo push is as follows:

(a) After the DSRC route is established,

(b) the push server transmits the URL, and

(c) the push client acquires the content using the HTTP.

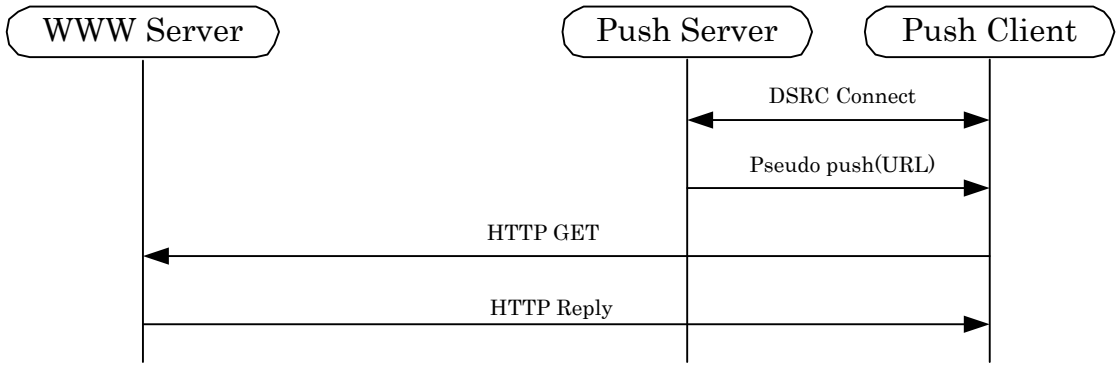


Figure 3.4-2 example of sequence of "pseudo push"

The push-type information delivery application defined in this document has the following functions.

The content push and pseudo push are realized by these functions.

The content push uses the functions (1), (2), (3) and (4), and the pseudo push uses the functions (1), (2) and (5).

(1) DSRC client resource acquisition function (Point-to-point communication)

(2) Push-type information delivery function

(a) Push-type information delivery without confirmation response (Point-to-point communication/broadcast communication)

(b) Push-type information delivery with confirmation response (Point-to-point communication)

(c) Abort of push-type information delivery with confirmation response (Point-to-point communication)

(3) Divide and send function in accordance with the DSRC client resource (Point-to-point communication)

(4) Pushed content replay request function (Point-to-point communication)

(5) Pseudo push function (Point-to-point communication)

In this document, the push server and push client indicates the push-type information delivery application mounted in the roadside system and OBE respectively, and the DSRC client indicates the push client, application which executes contents, and system equipped with such push client and application.

3.4.1.1 DSRC client resource acquisition function

In the push-type information delivery application interface, the push server acquires the following information as DSRC client resource information at initial connection:

(1) Maximum content size which can be received at a time by the push client (MaxPushBodySize)
(2) Maximum content size which can be handled by the DSRC client (MaxContentsSize)
(3) Content types which can be handled by the DSRC client
(4) Application types held by the DSRC client

The initial connection operation does not exist in broadcast communication. For executing services using broadcast communication, it is necessary to define in advance the minimum DSRC client resource in accordance with each content type. The minimum DSRC client resource in broadcast communication is outside the range of this document, and is not defined here.Refer to Subsection 3.4.6 for the application type and content type.

Reference:When the LPP is used, “MaxPushBodySize” is the maximum size of the LPP-SDU subtracted by the header size of the push-type information delivery application interface.

Reference:When the LPP is used, the division/assembly function of the LPP is used if the message size exceeds the MTU size in the LPCP in a service using broadcast communication.

3.4.1.2 Push-type information delivery function

In the push-type information delivery application, the push server specifies the content type and application type to be executed, and sends a content to the push client.

This protocol specifies the following two types of push operations:

(1) Push-type information delivery without confirmation response

The push server transmits a content to the push client. The push client does not return confirmation response to the push server. This operation is used in both point-to-point communication and broadcast communication.

(2) Push-type information delivery with confirmation response

The push server transmits content to the push client. When the push client receives the pushed content, the push client returns confirmation response to the push server.

The response sending timing can be selected among the following three types by the push server:

(1) When receiving of content is completed in the push client
(2) When transfer of the content to the application which will execute the content from the push client is completed
(3) When execution of the content is completed

This operation is available only in point-to-point communication because confirmation response to the push server is required.

In push-type information delivery with confirmation response, it is possible to abort the push operation from either the push client or the push server during the period from sending of the content to receiving of the response.

When the push server continuously sends contents, the receiving buffer may become insufficient in the push client, and contents may be overwritten or aborted. In such a case, the push client can notify the push server using the confirmation response at completion of transfer that it can receive the next content.

In the same push server, the 1-byte push ID is used to discriminate each push operation.

3.4.1.3 Devide and send function in accordance with DSRC client resource (OPTIONAL)

In the push-type information delivery application, the push server specifies the function to divide a send content into several segments and send segments. This function is provided to distribute content data larger than the receiving buffer size of the DSRC OBE.

In this function, the push server compares the content size (MaxPushBodySize) which can be received at a time by the push client on the client side acquired by the client resource acquisition function with the send content size (ContentSize).

If the comparison result is “MaxPushBodySize < ContentSize”, the push server divides the send content into several segments so that the send data size does not exceed “MaxPushBodySize”, and then sends segments.

The DSRC client not supporting this function should notify “MaxPushBodySize” and “MaxContentsSize” of a same size as the DSRC client resource information at initial connection.

A procedure example in the devide and send function is as follows:

(1) When the send content size is larger than “MaxPushBodySize" of the push client, the push server divides the send content into several segments so that the send data size does not exceed “MaxPushBodySize", sends the first segment, and then waits for receiving the next segment data send request from the push client. At this time, the push server makes valid the parameter value (isSegment) indicating that send data is a segment data. After that, the push server repeatedly sends 1-segment data every time the push client gives the next

segment data send request, and waits for the next request. When the send data is the final segment, the push server makes valid the parameter (isLast) indicating the end of division and transfer, and finishes the send processing.

(2) When received data is a segment data, the push client executes the processing (such as transfer to the external equipment) in accordance with the received data, and then gives the next segment data request to the push server at the timing at which the push client becomes ready for receiving the next segment data.The processing executed by the DSRC client during the period from receiving 1-segment data to giving next segment data send request is implementation-dependent. When received data is the final segment, the push client does not give the next segment data send request, but finishes the receiving processing.

In division and transfer, it is possible to abort the push operation from either the push client or the push server during the period from start of content sending to completion of content sending.

3.4.1.4 Pushed content replay request function

In the push-type information delivery application, the pushed content replay function is specified.

A procedure example to realize the replay request function is as follows:

(1) The push server makes valid the parameter (requireCache) indicating holding of content data in the push client, and sends content data.When the push client receives this message, the client holds received content. At this time, the push client links received content data with the push ID.

(2) The push server notifies the push client of the push ID of the content data sent in the step (1). The push client replays the content data corresponding to the notified push ID.

If the push ID overlaps between the received push operation and the cached content, the cached content is overwritten with the received content. The cached content data remains valid until it is overwritten with a received message having the same push ID or until it is aborted inside the DSRC client. The abort timing is implementation-dependent.

When same content is executed several times by the DSRC client using this function, the overall communication quantity can be reduced. Different applications can be specified in this function under consideration of the necessity to execute same content in different applications.

This function has two types, “without confirmation response” and “with confirmation

response" in the same way as the normal push-type information delivery function. This function is applicable only to point-to-point communication.

3.4.1.5 Pseudo push function

The pseudo push (smart pull) function is realized when the push server distributes the "SmartPull (SP)" type content.

As an example of the pseudo push function using the SP, the URL automatic distribution procedure at entrance into the communication area is offered as follows:

(1) After the DSRC client enters the communication area, the push server creates a command whose content type is "dsrc-smart-pull(129) and push body is SP defined separately, and transmits it to the push client with push-type information delivery without confirmation response or push-type information delivery with confirmation response.
(2) The DSRC client establishes IP communication.
(3) The push server acquires specified content using a protocol (such as HTTP) specified by the SP, and then displays it.

Reference:When DSRC-ASL/PPPCP is used for IP communication, the pseudo push distribution timing should interlock with a connection event of the PPPCP ("communication connection event" for the PPP line). The concrete realization method is implementation-dependent.

3.4.2 Definition of commands

3.4.2.1 Command system

Commands of the push-type information delivery application consist of the push operation command, confirmed push operation command, confirmed push response command, re-push operation command, re-confirmed push operation command, re-confirmed push response command, push abort operation command, next segment request command, next segment data distribution command and client information notice command.

Pseudo push services use pseudo push contents.

3.4.2.2 Command format

3.4.2.2.1 Push Operation Command

The push operation command "PushOperation" is used in push-type information delivery without confirmation response from the push server to the push client.

Table 3.4-1 shows the "PushOperation" format.

Table 3.4-1 the "PushOperation" command format.

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Command Type push(0)				RES	DC	RC	IS
2	pushID							
3	applicationType							
4	contentType							
5	contentsSize							
6								
7								
8								
9	Length of "pushBody"							
:	Contents of "pushBody"							

(1) Command Type

This field is set to an identifier "push(0)" to indicate the push operation command.

(2) RES

This field indicates the reservation area.

(3) DC

DC stands for “Duplicate Check”, and gives instruction to the push client to (or not to) execute the duplicate check. This field is set to “true (1)” for requesting the duplicate check, and is set to “false (0)” for not requesting the duplicate check.

This flag is valid only in broadcast communication, and is set to “0” in point-to-point communication.

(4) RC

RC stands for “RequireCache”, and gives instruction to the push client to (or not to) hold content data. This field is set to “true (1)” for requesting holding, and is set to “false (0)” for not requesting holding. This flag is valid only in point-to-point communication, and is set to “0” in broadcast communication.

(5) IS

IS stands for “IsSegment”, and indicates that content data is divided by the push devide and send function. This field is set to “true (1)” when content data is divided, and “false (0)” when content data is not divided.

(6) pushID

This field indicates the ID to identify the push operation.

(7) ApplicationType

This field indicates the application type which executes push content. Refer to Table 3.4-14 for details.

(8) contentType

This field indicates the content type of push content. Refer to Table 3.4-15 for details.

(9) contentsSize

This field indicates the push content data size in fixed 4-byte length in units of octet. The size of entire content before division should be set to when the devide and send function is used.

(10) Length of “pushBody”

This field indicates the length of subsequent content data in units of octet. This field is set to “0” when there is no subsequent content data. This field size is extended based on ASN.1 conding rules.

(11) Contents of “pushBody”

This field indicates content data of undefined length to be sent.

3.4.2.2.2 Confirmed Push Operation Command

The confirmed push operation command “ConfirmedPushOperation” is used in push-type information delivery with confirmation response from the push server to the push client.

Table 3.4-2 shows the “ConfirmedPushOperation” format.

Table 3.4-2 “ConfirmedPushOperation” command format.

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Command Type confirmed-push(1)				responseTiming		RC	IS
2	pushID							
3	applicationType							
4	contentType							
5	contentsSize							
6								
7								
8								
9	Length of “pushBody”							
:	Contents of “pushBody”							

(1) Command Type

This field is set to an identifier “confirmed-push(1)” to indicatre the confirmed push operation command.

(2) responseTiming

This field indicates the confirmation response return timing. Refer to Table 3.4-3 for details.

(3) RC

RC stands for “RequireCache”, and gives instruction to the push client to (or not to) hold content data. This field is set to “true (1)” for requesting holding, and is set to “false (0)” for not requesting holding. This flag is valid only in point-to-point communication, and is set to “0” in broadcast communication.

(4) IS

IS stands for “IsSegment”, and indicates that content data is divided by the push devide and send function.

(5) pushID

This field indicates the ID to identify the push operation.

(6) applicationType

This field indicates the application type which executes push content. Refer to Table 3.4-14 for details.

(7) contentType

This field indicates the content type of push content. Refer to Table 3.4-15 for details.

(8) contentsSize

This field indicates the push content data size in fixed 4-byte length in units of octet. The size of entire content before division should be set to when the divide and send function is used.

(9) Length of pushBody

This field indicates the length of subsequent content data in units of octet. This field is set to "0" when there is no subsequent content data. This field size is extended based on ASN.1 conding rules.

(10) Contents of pushBody

This field indicates content data of undefined length to be sent.

Table 3.4-3 Contents of "responseTiming"

Value	Identifier	Meaning
0	received	When receiving is completed in the OBE
1	transfered	When transfer to the outside is completed
2	executed	When execution of content is completed

3.4.2.2.3 Confirmed Push Response Command

The confirmed push response command "ConfirmedPushResponse" is used to return response from the push client which received the confirmed push operation command to the push server.

Table 3.4-4 shows the "ConfirmedPushResponse" format.

Table 3.4-4 "ConfirmedPushResponse" command format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Command Type confirmed-push-res(2)				RES			
2	pushID							

3	Length of “acknowledgement”
:	Content of “acknowledgement”

(1) Command Type

This field is set to an identifier “confirmed-push-res(2)” to indicate the confirmed push response command.

(2) RES

This field indicates the reservation area.

(3) pushID

This field indicate the ID of confirmed push operation command to identify the push operation.

(4) Length of “acknowledgement”

This field indicates the length of subsequent response data in units of octet. This field is set to “0” when there is no subsequent response data (default). This field size is extended based on ASN.1 conding rules.

(5) Content of “acknowledgement”

This field indicates the contents of response data of undefined length from the application which executes content. This field is used for individual service or future expansion, and its details are outside the range of specification described in this document. Additional information is not given when there is no specification (default value).

3.4.2.2.4 Re-push Operation Command

The re-push operation command “Re-PushOperation” is used in replay without confirmation response of content data distributed by the push operation command or confirmed push operation command response executed before from the push server to the push client.

Table 3.4-5 shows the “Re-PushOperation” format.

Table 3.4-5 “Re-PushOperation” command format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Command Type re-push(3)				RES			
2	pushID							
3	applicationType							

(1) Command Type

This field is set to an identifier “re-push(3)” to indicate the re-push operation command.

(2) RES

This field indicates the reservation area.

(3) pushID

This field is set to ID of the push operation to replay. This ID is set to the value that can identify push operation in the push server and the push client.

(4) applicationType

This field indicates the application type which executes push content. Refer to Table 3.4-14 for details.

3.4.2.2.5 Re-Confirmed Push Operation Command

The re-confirmed push operation command “Re-ConfirmedPushOperation” is used in replay with confirmation response of content data distributed by the push operation command or confirmed push operation command executed before from the push server to the push client.

Table 3.4-6 shows the “Re-ConfirmedPushOperation” format.

Table 3.4-6 “Re-ConfirmedPushOperation” command format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Command Type re-confirmed-push(4)				responseTiming		RES	
2	pushID							
3	applicationType							

(1) Command Type

This field is set to an identifier “re-confirmed-push(4)” to indicate the re-confirmed push operation command.

(2) responseTiming

This field indicates the confirmation response return timing. Refer to Table 3.4-3 for details.

(3) RES

This field indicates the reservation area.

(4) pushID

This field is set to ID of the push operation to replay. This ID is set to the value that

can identify push operation in the push server and the push client.

(5) applicationType

This field indicates the application type which executes push content. Refer to Table 3.4-14 for details.

3.4.2.2.6 Re-Confirmed Push Response Command

The re-confirmred push response command "Re-ConfirmedPush Response" is used when the push client which received the re-confirmed push operation command gives response to the push server.

Table 3.4-7 shows the "Re-ConfirmedPushResponse" format.

Table 3.4-7 "Re-ConfirmedPushResponse" command format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Command Type re-confirmed-push-res(5)				RES			
2	pushID							
3	Length of "acknowledgement"							
:	Contents of "acknowledgement"							

(1) Command Type

This field is set to an identifier "re-confirmed-push-res(5)" to indicate the re-confirmed push response command.

(2) RES

This field indicates the reservation area.

(3) pushID

This field indicates the ID of "Re-ConfirmedPushOperation" to identify the push operation.

(4) Length of "acknowledgement"

This field indicates the length of subsequent response data in units of octet. Stores "0" when there is no subsequent response data (default). This field size is extended based on ASN.1 conding rules.

(5) Content of "acknowledgement"

This field indicates the contents of response data of undefined length from the application which executes content. This field is used for individual service or future expansion, and its details are outside the range of specification described in this

document. Additional information is not given when there is no specification (default value).

3.4.2.2.7 Push Abort Operation Command

The push abort operation command "PushAbortOperation" is used to abort push delivery with confirmation response, re-push with confirmation response or push operation using the devide and send function, and used also to notify the push server of an error occurred in the push client which is executing push delivery with confirmation response, re-push with confirmation response or push operation using the devide and send function.

Table 3.4-8 shows the "PushAbortOperation" format.

Note:When aborting a push operation adopting a request/response type transaction of the LPP using the push abort operation command, issue the "Abort.req" primitive of the LPP and abort the transaction of the LPP at the same time.

Table 3.4-8 "PushAbortOperation" command format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Command Type push-abort(6)				RES			
2	pushID							
3	status							
4	Length of "supplementInfo"							
:	Contents of "supplementInfo"							

(1) Command Type

This field is set to an identifier "push-abort(6)" to indicate the push abort operation command.

(2) RES

This field indicates the reservation area.

(3) pushID

This field indicates the ID to identify the push operation.

(4) Status identifier "status"

This field indicates the reason why the push operation is aborted. Refer to Table 3.4-9 for details.

(5) Length of “supplementInfo”

This field indicates the data length of subsequent additional information in units of octet. This field is set to “0” when there is no subsequent response data. This field size is extended based on ASN.1 conding rules.

(6) Contents of “supplementInfo”

This field indicates data of undefined length up to 127 octets as the contents of additional information. Refer to Table 3.4-9 for details.

Table 3.4-9 Contents of status identifier "status"

Value	meanings	contents of supplementInfo
0	Prohibited from use	Nothing
1	PDU error (PDU structure error)	Nothing
2	PDU error (Undefined PDU)	Nothing
3	Push operation abort request*	Specified by the application which executes the content
4	Specified application type not supported	Nothing
5	Specified content type not supported	Nothing
6	Content improper*	Nothing
7	Content size error (Received size disagreed)	Nothing
8	Content size error (Maximum content size over)	Nothing
9	No content to be redistributed	Nothing
10	Devide and send error (Segment number sequence error)	Nothing
11	Devide and send error (Devide and send not supported)	Nothing
12-254	RES	
255	Other error codes	Nothing

* Note: Used in the push operation abort request from the application which executes the content.

3.4.2.2.8 Next Segment Request Command

The next segment request command “NextSegmentRequest” is used when the push client notifies the push server that it is ready for receiving the next segment data after it received a

segment data in delivery using the devide and send function.

Table 3.4-10 shows the “NextSegmentRequest” format.

Table 3.4-10 “NextSegmentRequest” command format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Command Type next-seg-request(7)				RES			
2	pushID							

(1) Command Type

This field is set to an identifier “next-seg-request (7)” to indicate the next segment request command.

(2) RES

This field indicates the reservation area

(3) pushID

This field indicates the ID to identify the push operation.

3.4.2.2.9 Next Segment Data Delivery Command

The next segment data delivery command “NextSegment” is used when the push server distributes the next segment data to the push client in delivery using the devide and send function.

Table 3.4-11 shows the “NextSegment” format.

Table 3.4-11 “NextSegment” format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Command Type nextSegment(8)				RES			isLast
2	pushID							
3	segmentNo							
4								
5	Length of “segmentBody”							
:	Contents of “segmentBody”							

(1) Command Type

This field is set to an identfier “nextSegment (8)” tp indicate the next segment data

delivery command.

(2) RES

This field indicates the reservation area

(3) isLast

This field indicates whether it is the final segment or not. This field is set to “true (1)” when “NextSegment” distributes the final segment. This field is set to “false (0)” for any other segment.

(4) pushID

This field indicates the ID to identify the push operation.

(5) segmentNo

This field indicates the segment sequence number. The sequence number is incremented in turn from “2”.

(6) Length of “segmentBody”

This field indicates the length of subsequent segment data in units of octet. This field size is extended based on ASN.1 conding rules.

(7) Contents of “segmentBody”

This field indicates the contents of segment data of undefined length.

3.4.2.2.10 Client Information Notice Command

The client information notice command “ClientInformation” is used when the push client notifies the push server of the functions held by the DSRC client at initial connection.

Table 3.4.12 shows the “ClientInformation” format.

Table 3.4-12 “ClientInformation” format

<table>
<tr><td></td><td>7(MSB)</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0(LSB)</td></tr>
<tr><td>1</td><td colspan="4">Command Type
clientInformation(15)</td><td colspan="4">version</td></tr>
<tr><td>2</td><td colspan="8" rowspan="2">applicationTypeList
(Stores the list of application types. Refer to Table 3.4-14 for details.)</td></tr>
<tr><td>:</td></tr>
<tr><td>:</td><td colspan="8">contentTypeList
(Stores the list of content types. Refer to Table 3.4-15 for details.)</td></tr>
<tr><td>:</td><td colspan="8" rowspan="3">maxPushBodySize</td></tr>
<tr><td>:</td></tr>
<tr><td>:</td></tr>
</table>

:	
:	maxContentsSize
:	
:	
:	
:	Length of “supplementInfo”
:	Contents of “supplementInfo”

(1) Command Type

This field is set to an identifier “clientInformation (15)” to indicate the client information notice command.

(2) version

This field indicates the application version of the push client. The current version is “0x01”.

(3) applicationTypeList

This field indicates the list of application types supported by the DSRC client.

(4) contentTypeList

This field indicates the list of content types supported by the DSRC client.

(5) maxPushBodySize

This field indicates the maximum content size which can be received at a time by the push client, in other words, the maximum size of the “pushBody” area of “PushOperation” and “ConfirmedPushOperation”, in units of octet.The area size of this identifier is fixed to 4 bytes.

(6) maxContentsSize

This field indicates the maximum content size which can be handled by the DSRC client in units of octet. The area size of this identifier is fixed to 4 bytes.

(7) Length of supplementInfo

This field indicates the data length of subsequent additional information in units of octet. This field is set to “0” when there is no subsequent response data. This field size is extended based on ASN.1 conding rules.

(8) Contents of supplementInfo

This field indicates the client information other than (1) to (7) above of undefined length up to 127 octets as the contents of additional information. This area is used for individual service or future expansion, and its details are outside the range of specification described in this document. Additional information is not given when there

is no specification (default value).

3.4.2.2.11 Pseudo push content

Pseudo push content is used to notify the content position (such as URL). And this content is set to in "pushBody" of "PushOperation" or "ConfirmedPushOperation", and distributed from the push server to the push client.

Table 3.4-13 shows the pseudo push content format.

Table 3.4-13 Psedu push content format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Length of "href"							
2	Contents of "href"							
:	:							
	Length of parameter							
	Contents of parameter							
	:							

(1) Length of "href"

This field indicates the data length of the information (URI) specifying the subsequent content position in units of octet. This field size is extended based on ASN.1 conding rules.

(2) Contents of "href"

This field indicates the contents of information (URI) of undefined length specifying the content position.

(3) Length of parameter

This field indicates the data length of subsequent parameter in units of octet. This field is set to "0" when there is no parameter. This field size is extended based on ASN.1 conding rules.

(4) Contents of parameter

This field indicates the contents of parameter of undefined length transferred to the URI indicated by "href".

3.4.3 The definition of data format

```
PushOperationCommand ::= CHOICE {
    push                    [0]     PushOperation,
    confirmed-push          [1]     ConfirmedPushOperation,
    confirmed-push-res      [2]     ConfirmedPushResponse,
    re-push                 [3]     Re-PushOperation,
    re-confirmed-push       [4]     Re-ConfirmedPushOperation,
    re-confirmed-push-res   [5]     Re-ConfirmedPushResponse,
    push-abort              [6]     PushAbortOperation,
    next-seg-request        [7]     NextSegRequest,
    nextSegment             [8]     NextSegment,
    dummy                   [9-14]  NULL,
    clientInformation       [15]    ClientInformation
}

PushOperation ::= SEQUENCE {
    res                 BIT STRING(SIZE(1)),    -- For future use
    duplicateCheck      BOOLEAN,
    requireCache        BOOLEAN,    -- TRUE(1) in specifying the content
  retention for the retry operation request
    isSegment           BOOLEAN,    -- TRUE(1) in segmentation transmission
    pushId              INTEGER(0..255),
    applicationType     ApplicationType,
    contentType         ContentType,
    contentSize         INTEGER(0..4294967295),
    pushBody            OCTET STRING -- content body for transmission
}

ApplicationType ::= CHOICE {
    default             [0]     NULL,
    browser             [1]     NULL,
    mailer              [2]     NULL,
    sound-player        [3]     NULL,
```

```
    video-player            [4]         NULL,
    tts                     [5]         NULL,
    mobile-device-browser   [6]         NULL,
    store                   [7]         NULL,
    vics                    [8]         NULL,
    text-display            [9]         NULL,
    safety                  [10]        NULL,
    image-display           [11]        NULL,
    payment                 [12]        NULL,
    other                   [13-254]    NULL,
    private                 [255]       OCTET STRING
}

ContentType ::= CHOICE{
    everyType               [0]         OCTET STRING,
    text                    [1]         OCTET STRING,
    text-plain              [2]         NULL,
    text-enrich             [3]         NULL,
    text-html               [4]         NULL,
    text-xml                [5]         NULL,
    text-x-hdml             [6]         NULL,
    text-x-html             [7]         NULL,
    text-tts                [8]         NULL,
    otherTextType           [9-15]      NULL,
    image                   [16]        OCTET STRING,
    image-jpeg              [17]        NULL,
    image-gif               [18]        NULL,
    image-bmp               [19]        NULL,
    image-tiff              [20]        NULL,
    image-png               [21]        NULL,
    otherImageType          [22-31]     NULL,
    audio                   [32]        OCTET STRING,
    audio-wav               [33]        NULL,
    audio-mp3               [34]        NULL,
    audio-wma               [35]        NULL,
```

audio-aiff	[36]	NULL,
audio-midi	[37]	NULL,
audio-adpcm	[38]	NULL
audio-celp	[39]	NULL
otherAudioType	[40-46]	NULL,
audio-encoded-voice-type1	[47]	NULL
video	[48]	OCTET STRING,
video-mpeg	[49]	NULL,
video-real	[50]	NULL,
video-qt	[51]	NULL,
video-wmv	[52]	NULL,
reservedForFutureVideoType	[53-63]	NULL,
message	[64]	OCTET STRING,
otherMessageType	[65-79]	NULL,
application	[80]	OCTET STRING,
application-java-vm	[81]	NULL,
application-postscript	[82]	NULL,
othereAppType	[83-95]	NULL,
multipart	[96]	OCTET STRING,
otherMultiPartType	[97-127]	NULL,
dsrc	[128]	OCTET STRING,
dsrc-smart-pull	[129]	NULL,
dsrc-vics	[130]	NULL,
dsrc-mime	[131]	NULL,
dsrc-safety	[132]	NULL,
dsrc-multipart	[133]	NULL,
dsrc-privateSpot_text_plain	[134]	NULL,
dsrc-privateSpot_image_jpeg	[135]	NULL,
dsrc-privateSpot_image_gif	[136]	NULL,
dsrc-privateSpot_image_bmp	[137]	NULL,
dsrc-privateSpot_image_tiff	[138]	NULL,
dsrc-privateSpot_image_png	[139]	NULL,
otherType	[140-239]	NULL,
private	[240-255]	NULL

}

```
ConfirmedPushOperation ::= SEQUENCE {
    responseTiming      ResponseTiming,
                            -- specify the response timing of acknowledge
    requireCache        BOOLEAN,
                            -- TRUE(1) in requesting the data retention
    isSegment           BOOLEAN,
                            -- TRUE(1) in segmentation transmission
    pushId              INTEGER(0..255),
    applicationType     ApplicationType,
    contentType         ContentType,
    contentSize         INTEGER(0.. 4294967295),
    pushBody            OCTET STRING
}

ResponseTiming ::= INTEGER{
    received    (0),    -- on completion of OBE reception
    transfered  (1),    -- on completion of transfer for external
    executed    (2)     -- on completion of execute a content
}(0..3)

ConfirmedPushResponse ::= SEQUENCE {
    res                 BIT STRING(4),          -- For future use
    pushId              INTEGER(0..255),        -- corresponding Push ID
    acknowledgement     OCTET STRING
}

Re-PushOperation ::= SEQUENCE {
    res                 BIT STRING(SIZE(4)),    -- For future use
    pushId              INTEGER(0..255),
                                    -- target Push ID for retry operation
    applicationType     ApplicationType
}

Re-ConfirmedPushOperation ::= SEQUENCE {
```

```
    responseTiming      ResponseTiming,
                                -- specify the response timing of acknowledge
    res                 BIT STRING(SIZE(2)),    -- For future use
    pushId              INTEGER(0..255),
                                -- target Push ID for retry operation
    applicationType     ApplicationType
}

Re-ConfirmedPushResponse ::= SEQUENCE {
    res                 BIT STRING(SIZE(4)),    -- For future use
    pushId              INTEGER(0..255),        -- operated Push ID again.
    acknowledgement     OCTET STRING
}

PushAbortOperation ::= SEQUENCE {
    res                 BIT STRING(SIZE(4)),    -- For future use
    pushId              INTEGER(0..255),        -- target Push ID
    status              INTEGER(0..255),        -- status code
    supplementInfo      OCTET STRING(SIZE(0..255))
                                                -- supplement information
}


NextSegRequest ::= SEQUENCE {
    res                 BIT STRING(SIZE(4)),    -- For future use
    pushId              INTEGER(0..255)
}

NextSegment ::= SEQUENCE{
    res                 BIT STRING(SIZE(3)),    -- For future use
    isLast              BOOLEAN,                -- TRUE(1) in a last segment
    pushId              INTEGER(0..255),        -- corresponding Push ID
    segmentNo           INTEGER(0..65535),      -- sequence number
    segmentBody         OCTET STRING            -- divided pushBody
}
```

```
ClientInformation ::= SEQUENCE {
    version                 INTEGER(0..15),          -- value is set up to 1 at first
    applicationTypeList     ApplicationTypeList,
    contentTypeList         ContentTypeList,
    maxPushBodySize         INTEGER(0..4294967295),
    maxContentsSize         INTEGER(0..4294967295),
    supplementInfo          OCTET STRING(SIZE(0..255)) -- supplement information
}

ApplicationTypeList ::= SEQUENCE OF ApplicationType

ContentTypeList ::= SEQUENCE OF ContentType

SP ::= SEQUENCE{
    href                    URI,
    parameter               OCTET STRING
}

URI ::= OCTET STRING
```

3.4.4 Relationship with Other Standards

- Use AID=18 of DSRC APPLICATION SUB LAYER.
- Use Local Port Control Protocol of NCP in ASL
- Use 0x0C0A for the local port number.
- Use the following types of transaction services of the local port protocol:
  (1) When not using the push devide and send processing
  (a) Use a unidirectional data send transaction for push without confirmation response and replay without confirmation response.
  (b) Use a request/response type transaction for push with confirmation response and replay with confirmation response.
  (2) When using the push devide and send processing
  (a) Use a request/response type transaction for push with confirmation response and push without confirmation response for any segment other than the final segment.
  (b) For the final segment, use a unidirectional data send transaction for push without confirmation response, and use a request/response type transaction for push with confirmation response.
  (3) When transferring consecutively a same content in broadcast communication, use the transaction re-execution function of the LPP.
  (4) When the message size exceeds the MTU size in the LPCP, use the division/assembly function of the LPP.
  (5) When aborting a push operation adopting a request/response type transaction using the push abort operation command, issue the “Abort.req" primitive of the LPP and abort the transaction of the LPP at the same time.

- Use the following types of primitives of the local port protocol:
  (1) Send each of the “PushOperation”, “ConfirmedPushOperation”, “Re-PushOperation”, “Re-ConfiremdPushOperation”, “NextSegment” and “ClientInformation” commands using the “Invoke.req” primitive.
  (2) Send each of the “ConfirmedPushResponse”, “Re-ConfirmationPushResponse” and “NextSegmentRequest” commands using the “Invoke.res” primitive.
  (3) Use the “Invoke.res” primitive for sending the “PushAbortOperation” command when aborting a push operation adopting a request/response type transaction from the OBE side, and use the “Invoke.req” primitive (TT = 0) in any other case.

3.4.5 Procedures

3.4.5.1 Procedure using client resource acquisition function

(1) After the DSRC is connected and communication from the push client is enabled, the push client creates the client information notice command “ClientInformation”, and notifies the push server of the client information.

Figure 3.4-3 shows a sequence example of the client resource acquisition function.

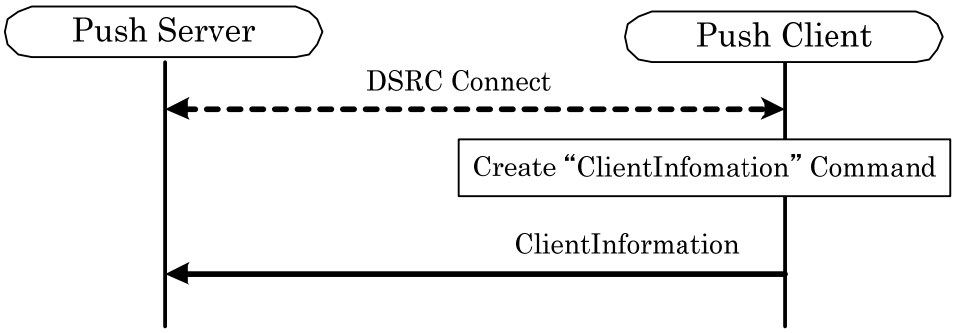


Figure 3.4-3 example of the client resource acquisition function

3.4.5.2 Push-type information delivery procedures (without division)

3.4.5.2.1 Data transfer procedure in push type information delivery without confirmation response

(1) The push server creates the push operation command “PushOperation”, and distributes a content to the push client.

(2) When the push client receives “PushOperation” sent in the step (1), the push client executes the processing for the received content in accordance with the content type and application type specified by “contentType” and “applicationType”.

Figure 3.4-4 shows an example of the data transfer procedure sequence in push delivery without confirmation response (without devide and send).

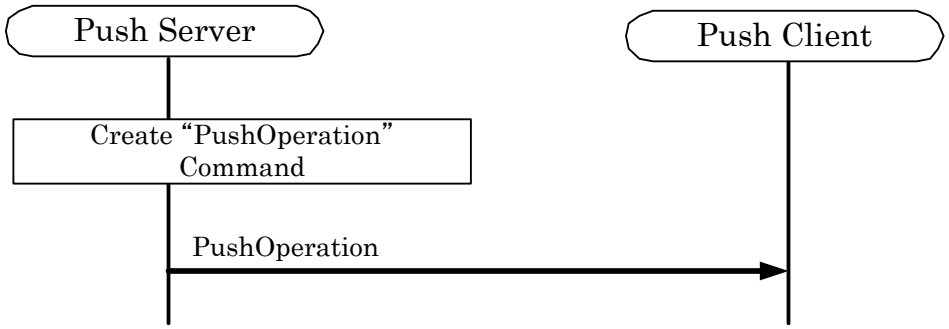


Figure 3.4-4 example of the data transfer procedure sequence in push type information delivery without confirmation response (without devide and send)

3.4.5.2.2 Data transfer procedure in push type information delivery with confirmation response

(1) The push server creates the confirmed push operation command “ConfirmedPushOperation”, and distributes a content to the push client.

(2) When the push client receives “ConfirmedPushOperation” sent in the step (1), the push client executes the processing for the received content in accordance with the content type and application type specified by “contentType” and “applicationType”.

(3) The push client creates the confirmed push response command “ConfirmedPushResponse” at the timing specified by the “responseTiming” parameter, and sends it to the push server.

When the received “ConfirmedPushOperation” corresponds to any of the following, the push client sends the corresponding push abort operation command to the push server, and terminates the processing.

(a) When the DSRC client does not support the specified content type, the push client creates the push abort operation command “PushAbortOperation” whose status identifier indicates “Specified content type not supported”, and sends it to the push server.

(b) When the DSRC client does not support the processing in accordance with the specified application type, the push client creates the push abort operation command “PushAbortOperation” whose status identifier indicates “Specified application type not supported”, and sends it to the push server.

(c) When the content size specified in the received “ConfirmedPushOperation” disagrees with the actually received content size, the push client creates the push abort operation command “PushAbortOperation” whose status identifier indicates “Content size error (Received size disagreed)”, and sends it to the push server.

Figure 3.4-5 shows an example of the data transfer procedure sequence in push-type information delivery with confirmation response (without devide and send).

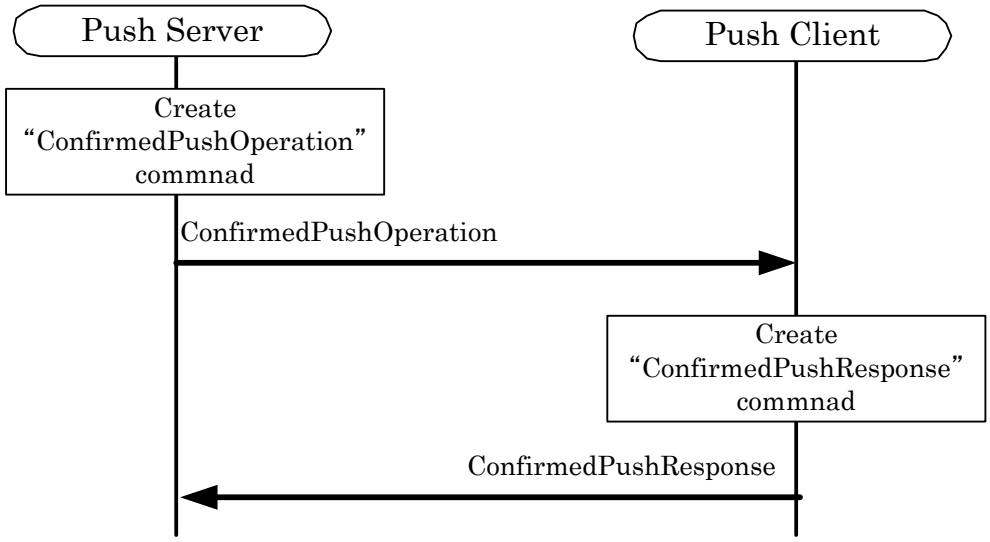


Figure 3.4-5 example of the data transfer procedure sequence in push type information delivery with confirmation response (without devide and send)

3.4.5.2.3 Data transfer procedure in push type information delivery using broadcast communication

(1) The push server creates one or more push operation commands “PushOperation” in accordance with a request from the service application, and distributes contents. At this time, if the service application does not specify the use of repeated send function, the push server sets “0” to the DC flag, and sends the created “PushOperation”. If the service application specifies the use of repeated send function, the push server sets “1” to the DC flag, and sends the created “PushOpeartion” repeatedly.Refer to Attached data G2.1 (b) for consideration about push ID assignment in broadcast communication.

(2) When the mobile station enters the DSRC communication area, the push client receives “PushOperation” sent in the step (1).The push client holds its push ID if the DC flag value is “1”.

(3) The push client executes the processing for the received content in accordance with the content type and application type specified by “contentType” and “applicationType”.

(4) When the push client receives “PushOperation” whose DC flag value is “1” and push ID remains held in the same communication area, the client aborts the received “PushOperation”.

Note:When the LPP sends the DSRC disconnection notice to the push client, the push client shall annuled the push ID stored if the DC flag value is “1”. The push client can store up to 128 push IDs. If the push client receives 129 or more push IDs, it shall annuled the push ID in turn from the one stored earliest.

Reference:When a same push ID is sent consecutively using the LPP in the step (1), the replay function of the transaction is used. Accordingly, the duplicate received data in the step (4) is annuled by the LPP, and is not given to the push-type information delivery application.

Figure 3.4-6(a) shows an example of the data transfer procedure sequence in push type broadcast delivery using the repeated send function, and Fig. 3.4-6(b) shows an example of the data transfer procedure sequence in push type delivery using broadcast communication without using the repeated send function.

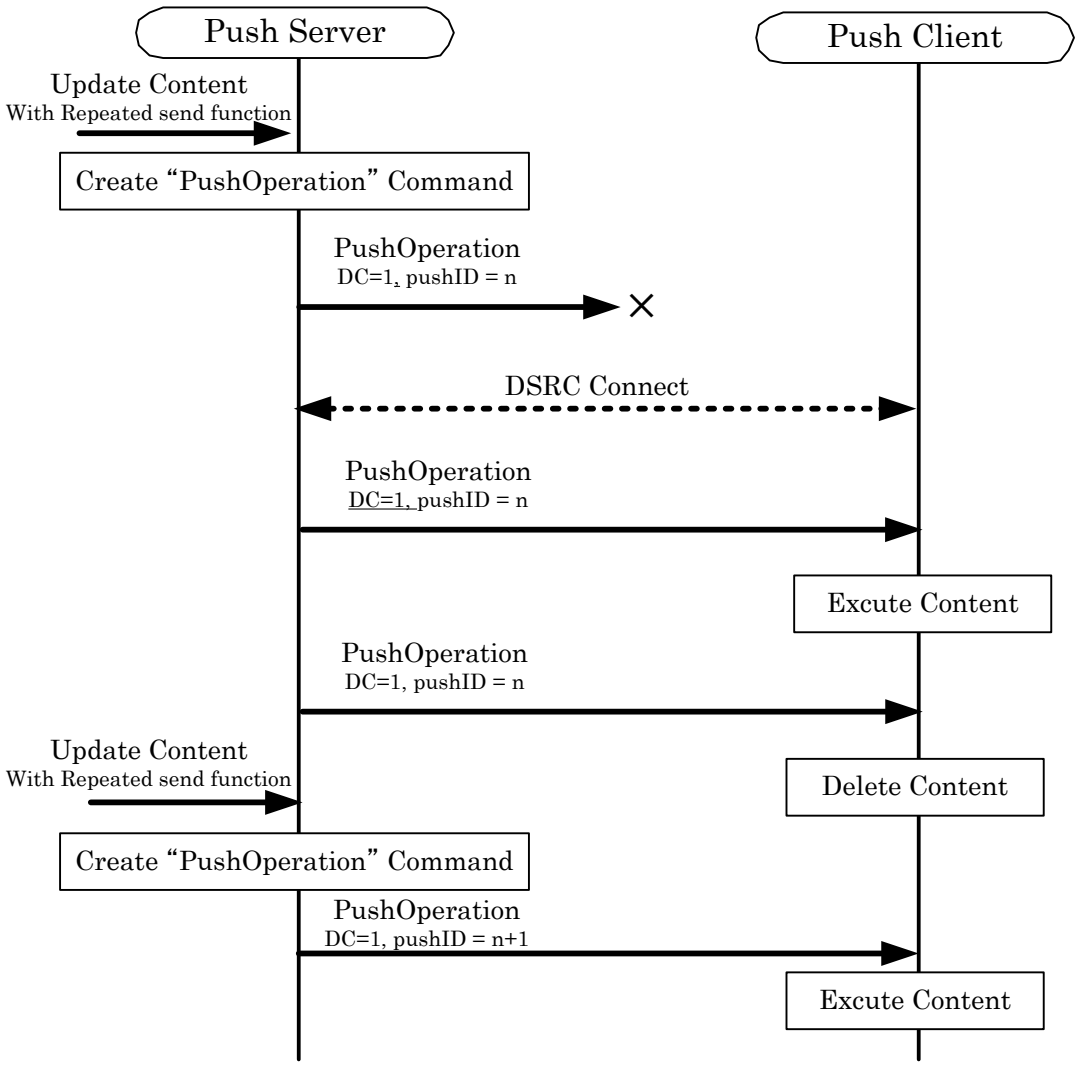


Figure 3.4-5 (a) example of the data transfer procedure sequence in push type information delivery using broadcast communication (with repeated send function)

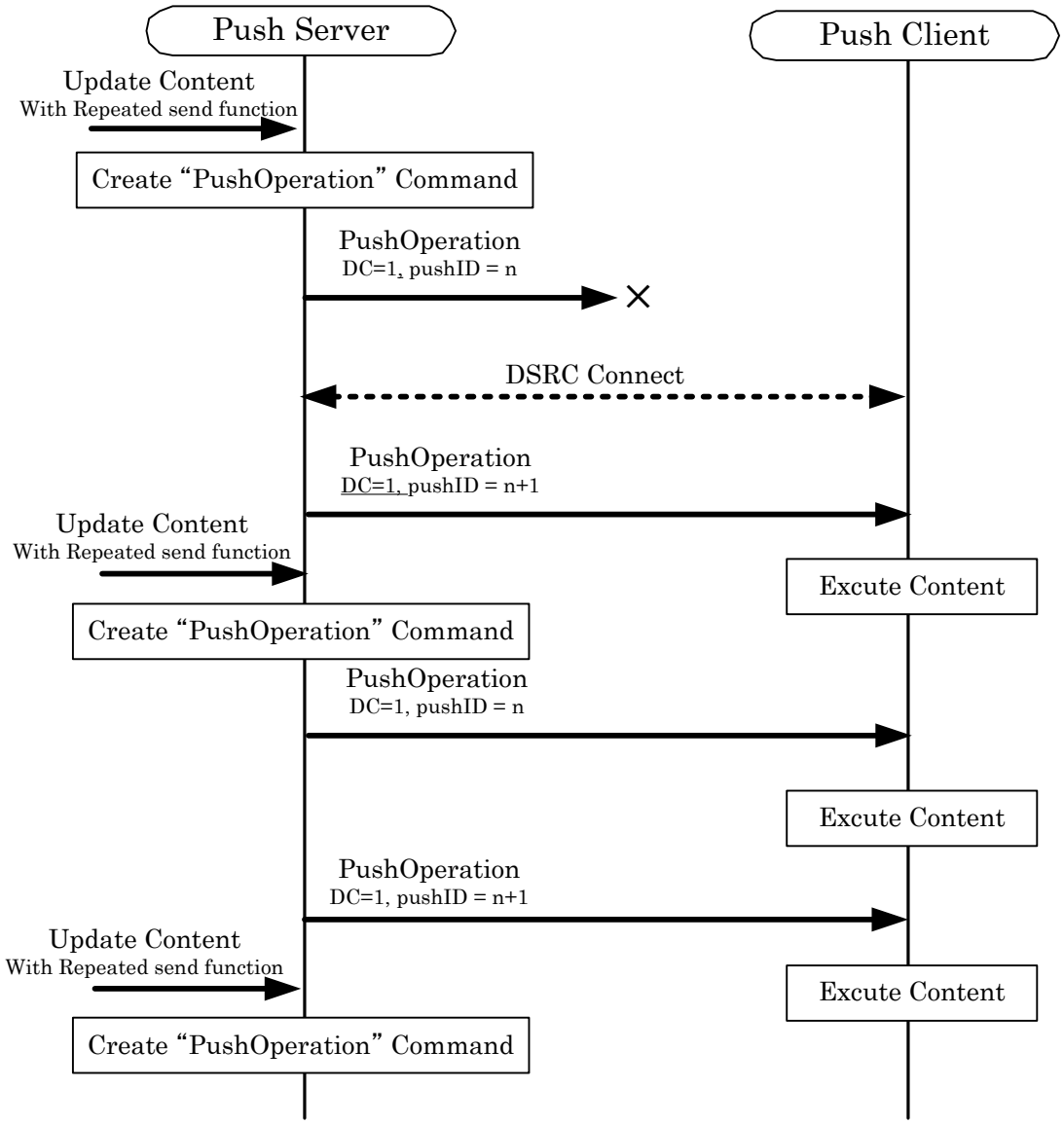


Figure 3.4-5 (b) example of the data transfer procedure sequence in push type information delivery using broadcast communication without the repeated send function

3.4.5.3 Data transfer procedure using devide and send function

(1) This transaction is started when the push server sends a content whose size exceeds “maxPushBodySize”.

(2) The push server divides a content from the head by “maxPushBodySize”, creates the push operation command “PushOperation” or confirmed push operation command “ConfirmedPushOperation” whose “pushBody” is a segment data, and sends it to the push client. At this time, the push server acquires “maxPushBodySize” in advance using the client resource acquisition function, and sets the IS flag indicating devide and send.

(3) When the push client receives “PushOperation” or “ConfirmedPushOperation” sent in the step (2), the push client transfers the received data to the external terminal in accordance with the content type and application type specified by “contentType” and “applicationType”.

(4) When the push client becomes ready for receiving the next segment, the push client creates the next segment request command “NextSegmentRequest”. And the push client sends it to the push server.

(5) When the push server receives “NextSegmentRequest” sent in the step (4) or (6), the push server divides the unsent portion of the content by “maxPushBodySize” from the head, creates the next segment data delivery command “NextSegment” whose "segmentBody” is a segment data, and then sends it to the push client. At this time, the push server assigns the segment number “2” to the first “NextSegment”, and assigns an incremented segment number to the second and later “NextSegment” in turn. The push server sets the “isLast” flag to the send data which is the final segment data.

(6) When the push client receives "NextSegment" sent in the step (5), the push client transfers the received data to the external terminal same as the step (3). If the “isLast” flag is not set in the received data, the push client creates the next segment request command “NextSegmentRequest” after completing the transfer processing, and sends it to the push server. If the “isLast” flag is set in the received data, the push client terminates the transaction after completing the transfer processing in the case of “PushOperation”. In the case of “ConfirmedPushOperation, the push client creates the confirmed push response command “ConfirmedPushResponse” at the timing specified by the “responseTiming” parameter in “ConfirmedPushOperation” received in the step (2), and sends it to the push server.

When a command received by the push client corresponds to any of the following, the push client sends the corresponding push abort operation command to the push server, and terminates the processing.

(a) When the DSRC client does not support the specified content type in the step (3), the push client creates the push abort operation command “PushAbortOperation” whose status identifier indicates “Specified content type not supported”, and sends it to the push server.
(b) When the DSRC client does not support the processing in accordance with the specified application type in the step (3), the push client creates the push abort operation command “PushAbortOperation” whose status identifier indicates “Specified application type not supported”, and sends it to the push server.
(c) When the segment number specified by “NextSegment” received in the step (6) is wrong, the push client creates the push abort operation command “PushAbortOperation” whose status identifier indicates “Devide and send error (Segment number sequence error)”, and sends it to the push server.
(d) When the total received content size exceeds “maxContentsSize” in the step (6), the push client creates the push abort operation command “PushAbortOperation” whose status identifier indicates “Content size error (Maximum content size over)”, and sends it to the push server.
(e) If the content size specified in “ConfirmedPushOperation” received in the step (3) disagrees with the actually received content size when receiving “NextSegment” whose “isLast” flag is set in the step (6), the push client creates the push abort operation command “PushAbortOperation” whose status identifier indicates “Content size error (Received size disagreed)”, and sends it to the push server.

Figures 3.4-7 and 3.4-8 show examples of the data transfer procedure sequence in push-type information delivery without confirmation response and push-type information delivery with confirmation response using the devide and send function.

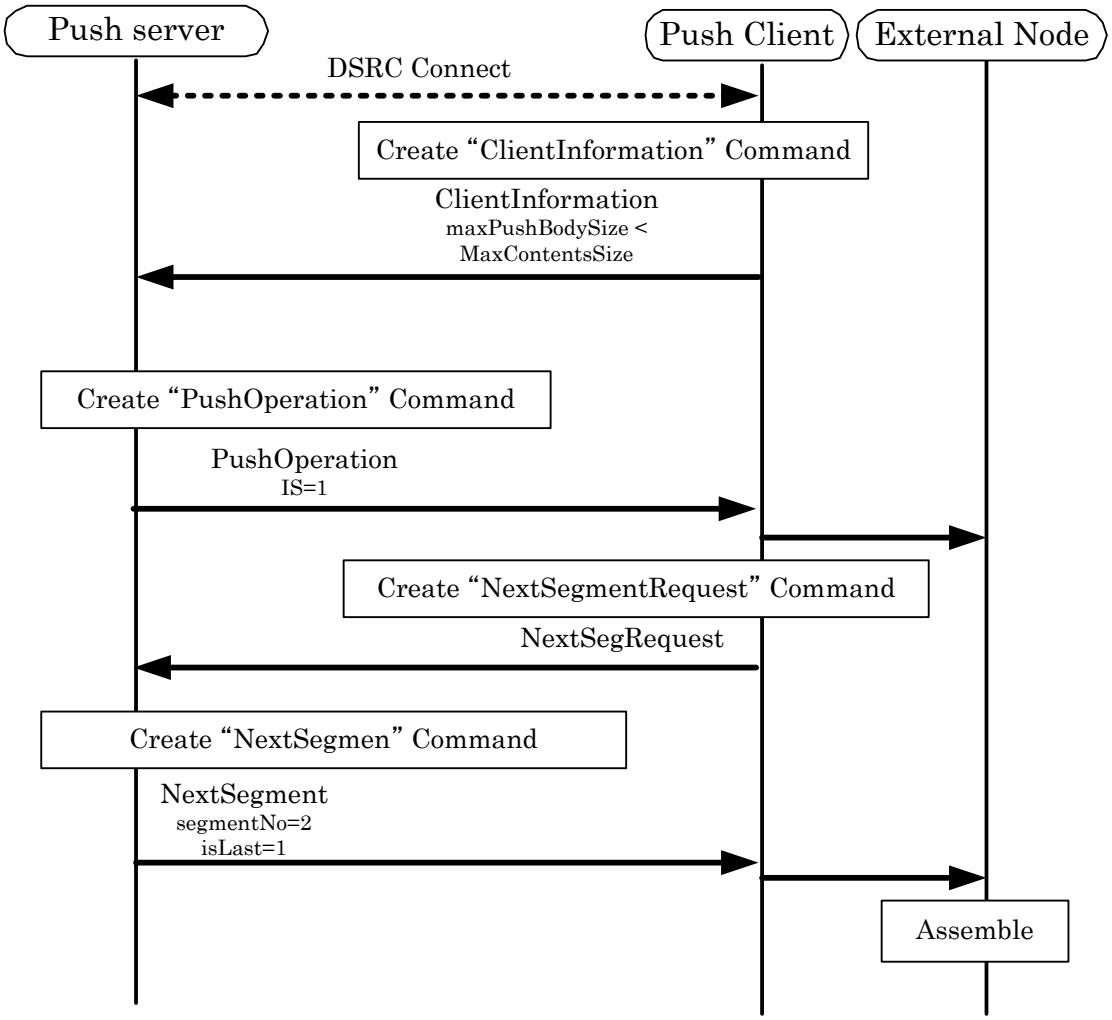


Figure 3.4-7 examples of the data transfer procedure sequence in push-type information delivery without confirmation response

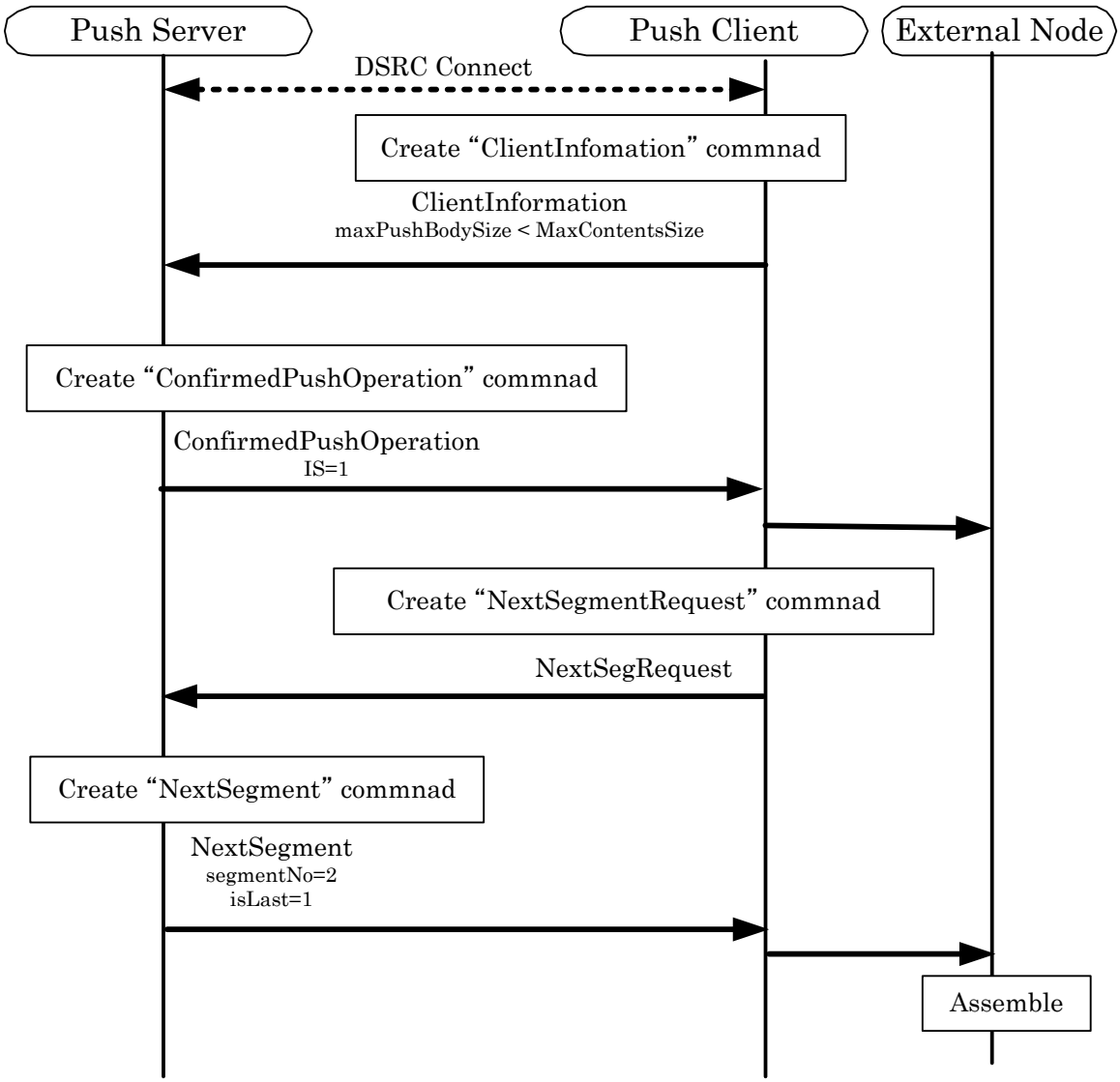


Figure 3.4-8 examples of the data transfer procedure sequence in push-type information delivery with confirmation response with the devide and send function.

3.4.5.4 Data transfer procedure using pushed content replay request function

3.4.5.4.1 Re-push function without confirmation response

(1) The push server distributes a content to the push client using the push operation command “PushOperation” or confirmed push operation command “ConfirmedPushOperation”. At this time, the push server sets the RC flag.

(2) When the push client receives “PushOperation” or “ConfirmedPushOperation” sent in the step (1), the push client executes and saves the received content.If a content cached in the past has the same ID, the push client overwrites the existing cached data.

(3) The push server creates the re-push operation command “Re-PushOperation” using the same ID as “pushID” in PushOperation” or “ConfirmedPushOperation” sent in the step (1), and sends it to the push client.

(4) When the push client receives “Re-PushOperation” sent in the step (3), the push client reads the content saved in the step (3), and executes the read content using the method specified by “applicationType”.

Figure 3.4-9 shows an example of the data transfer procedure sequence using the re-push function without confirmation response.

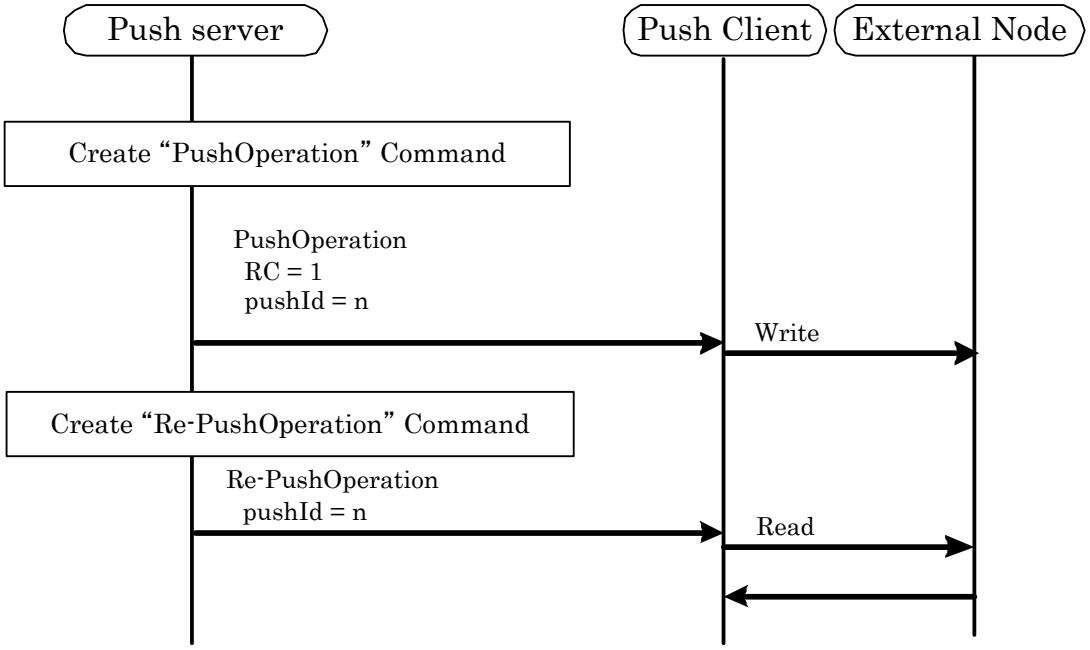


Figure 3.4-9 an example of the data transfer procedure sequence using the re-push function without confirmation response

3.4.5.4.2 Re-push function with confirmation response

(1) The push server distributes a content to the push client using the push operation command "PushOperation" or confirmed push operation command "ConfirmedPushOperation". At this time, the push server sets the RC flag.

(2) When the push client receives "PushOperation" or "ConfirmedPushOperation" sent in the step (1), the push client executes and saves the received content. If a content cached in the past has the same ID, the push client overwrites the existing cached data.

(3) The push server creates the re-push operation command with confirmation response "Re-ConfirmedPushOperation" using the same ID as "pushID" in "PushOperation" or "ConfirmedPushOperation" sent in the step (1), and sends it to the push client.

(4) When the push client receives "Re-ConfirmedPushOperation" sent in the step (3), the push client reads the content saved in the step (3), and executes the read content using the method specified by "applicationType".

(5) The push client creates the re-confirmed push response command "Re-ConfiremdPushResponse" at the timing specified by the "responseTiming" parameter, and sends it to the push server.

When a command received by the push client corresponds to any of the following, the push client sends the corresponding push abort operation command to the push server, and terminates the processing.

(a) When the content of the specified push ID is not cached in the step (4), the push client creates the push abort operation command “PushAbortOperation” whose status identifier indicates “No content to be distributed”, and sends it to the push server.

(b) When the DSRC client does not support the processing in accordance with the specified application type in the step (4), the push client creates the push abort operation command “PushAbortOperation” whose status identifier indicates “Specified application type not supported”, and sends it to the push server.

Figure 3.4-10 shows an example of the data transfer procedure sequence using the re-push function with confirmation response.

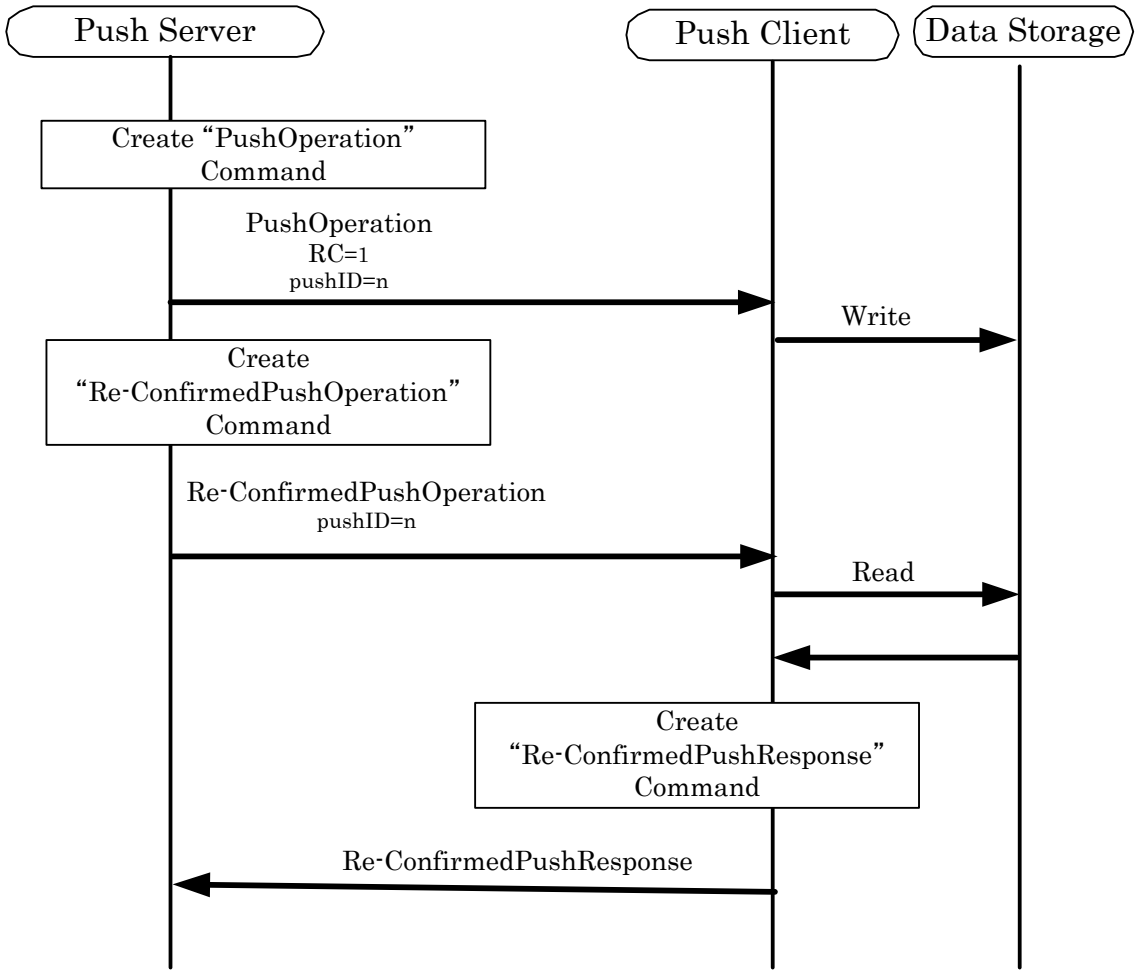


Figure 3.4-10 an example of the data transfer procedure sequence using the re-push function with confirmation response

3.4.6 Application type and content type list

Table 3.4-14 Application type list

Application	Identifier	Value	Remarks
default	default	0x00	
Web Browser	browser	0x01	
Mailer	mailer	0x02	
Sound Player	sound-player	0x03	
Movie Player	video-player	0x04	
TTS	tts	0x05	
Mobile Browser	mobile-device-browser	0x06	
Store	store	0x07	
VICS	vics	0x08	
Text display	text-display	0x09	
Driving Safety Support	safety	0x0A	
Image Display	image-display	0x0B	
Payment	payment	0x0C	
Others	others	0x0D-0xFE	
Any	private	0xFF	

Table 3.4-15 Content type list

Contents Type	Value	fprmat of pushBody	Remarks
/	0x00		(Note 1)
text/*	0x01		(Note 1)
text/plain	0x02	Shift-jis	
text/enrich	0x03	Enrich	Enrichtext
text/html	0x04	HTMLfile	HTMLtext
text/xml	0x05	XMLfile	XMLtext
text/x-hdml	0x06	X-HDMLfile	X-HDMLtext
text/x-html	0x07	X-HTMLfile	X-HTMLtext
text/tts	0x08	TTSfile	JEITA TT-6004
otherTextType	0x09-0x0F	-	(Note 2)
image/*	0x10	Image file	(Note 1)
image/jpeg	0x11		jpeg file
image/gif	0x12		gif file
image/bmp	0x13		bmpfile
image/tiff	0x14		tifffile
image/png	0x15		pngfile
otherImageType	0x16-0x1F	-	(Note 2)
audio/*	0x20	Audio file	(Note 1)
audio/wav	0x21		WAV file
audio/mp3	0x22		MP3file
audio/wma	0x23		WMAfile
audio/aiff	0x24		AIFFfile
audio/midi	0x25		MIDIfile
audio/adpcm	0x26		IMA-ADPCMfile
audio/celp	0x027		CELP encoded file
otherAudioType	0x28-0x2E	-	(Note 2)
audio/encoded-voice-type1	0x2F	Audio file	(Note 2)
video/*	0x30	Movie file	(Note 1)
video/mpeg	0x31		MPEGfile
video/real	0x32		RealPlayerfile
video/qt	0x33		QuickTimefile

video/wmv	0x34		WMVfile
otherVideoType	0x35-0x3F		(Note 2)
message/*	0x40	mail message defined RFC822	(Note 1)
otherMessageType	0x41-0x4F	-	(Note 2)
application/*	0x50	application data files	(Note 1)
application/java-vm	0x51		Java Virtual Machine
application/postscript	0x52		PostScript
otherApplicationType	0x53-0x5F	-	(Note 2)
multipart/*	0x60	Multi-part message	(Note 1)
otherMultipartType	0x61-0x7F	-	(Note 2)
dsrc/*	0x80	Content for DSRC applications	(Note 1)
dsrc/smart-pull	0x81	Refet to 3.4.3	
dsrc/vics	0x82		data format for VICS service
dsrc/mime	0x83	MIME encoded textfile	
dsrc/safety	0x84		
dsrc/multipart	0x85	multipart contents format	Refer to JEITA TT6003
dsrc/privateSpot_text_plain	0x86	Shift JIS line feed code CR+LF	Refer to Guideline for In-vehicle Display Systems
dsrc/privateSpot_image_jpeg	0x87	Image File Binary Format	
dsrc/privateSpot_image_gif	0x88		
dsrc/privateSpot_image_bmp	0x89		
dsrc/privateSpot_image_tiff	0x8A		
dsrc/privateSpot_image_png	0x8B		
otherType	0x8C-0xEF		(Note 2)
private	0xF0-FF		

Note 1: The content type specification method by “octet string” utilizes the “Content-Type” header field specified in RFC2045.

Note 2: Number assignment to areas not assigned in specific contents/applications is outside the range of specification described in this document.

3.5 OBE ID Communication Application

3.5.1 Outline of Functions

3.5.1.1 Functions

The OBE ID communication application notifies the roadside system of the ID held by the OBE so that the roadside system can identify the OBE.

The OBE ID communication application has the function to notify the OBE ID and the maintenance function to register, delete, etc. the OBE ID.

The OBE ID is registered in the nonvolatile memory, etc. of the OBE, and the recorded information is assured even if the power is turned OFF.

Maintenance of the OBE ID is available on the assumption that proper access control is achieved in the OBE, has arbitrary specification determined by the manufacturer (maybe including use of Vehicle navigation equipment or use of the HMI incorporated in the OBE), and is outside the range of specification described in this document.

3.5.1.2 OBE ID registration information

The OBE ID registration information “ObuIDForRegistration” (see Subsection 3.5.3) consists of the acquirer ID, ID condition and OBE ID.

3.5.1.3 Acquirer ID

The acquirer ID (ApplicationServiceProvider) identifies the provider who registers and acquires the OBE ID for the OBE.

In acquirer ID assignment, two methods, the representative assignment method in which upper 2 bytes indicate the representative number and lower 6 bytes indicate the sub number and the individual assignment method in which all of 8 bytes indicate a number, can be used together.

3.5.1.4 ID condition

The ID condition (IDCondition) indicating the ID condition specified by the registrant can be set for each acquirer. Table 3.5-1 shows the contents of "IDCondition". The security executed individually inside each application is outside the range of specification described in this document.

Table 3.5-1: Contents of "IDCondition"

Field name	True (1)	False (0)	Remarks
plaintextIDRefusal	Refuses ID send in plain text.	Permits ID send in plain text.	
ciphertextIDRefusal	Refuses ID send in cipher text.	Permits ID send in cipher text.	For individual in-application security
mutualAuthentication	Requires mutual authentication.	Does not require mutual authentication.	For individual in-application security
userApproval	Requires approval by user at each send.	Does not require approval by user.	(Reserved)
idUnlock	Enables deletion of this ID.	Disables deletion of this ID.	
spf	Permits ID send in SPF cipher.	Does not permit ID send in SPF cipher.	For common SPF security
fill	-	-	Filling with "0"

Note:When not using the individual in-application security (default), set "ciphertextIDRefusal" to "1" and "mutualAuthentication" to "0".Refer to Attached data B for handling using the common SPF.

3.5.1.5 OBE ID

The OBE ID uniquely identifies the OBE when combined with the acquirer ID.

The OBE ID is assigned in units of provider (acquirer) ID. One OBE ID should be used only once in each provider.

It is possible that single OBE has contracts with one or more service providers having different acquirer IDs. In such a case, different OBE ID is assigned to each acquirer ID in single OBE.

Accordingly, it is recommended that one or more OBE IDs can be registered for single OBE.

Reference:The OBE ID can be assigned in the following methods:

(a) Representative assignment method

Upper 2 bytes indicate the representative number (“manufacturerID (0 … 65535)” used in the VST), and lower 6 bytes indicate the sub number.

(b) Individual assignment methods

All of 8 bytes indicate a number.

3.5.2 Command

3.5.2.1 Command System

Commands in the OBE ID communication application consist of normal commands, OBU denial response command from the OBE, maintenance commands and authentication commands.

The normal commands consist of “ID 1st request command, ID 2nd request command, ID 1st response command, ID 2nd response command, end notice command and end response command”. The maintenance commands consist of “ID registration request command, ID registration response command, registered ID deletion request command, registered ID deletion response command, registered ID list request command, registered ID list response command, ID condition change request command and ID condition change response command”. The version number is “1” for the OBE ID communication application described in this document.

Authentication commands are used for individual authentication inside the OBE ID communication application, and details are outside the range of specification described in this document.

For acquiring the OBE ID through communication between the roadside system and the Vehicle, the roadside system notifies the OBE of the acquirer ID, and the OBE gives the OBE ID in accordance with the acquirer ID.

3.5.2.2 Command Format

3.5.2.2.1 Normal Commands

3.5.2.2.1.1 ID 1st Request Command

The ID 1st request command is used when the roadside system asks the OBE at the first time to give the OBE ID. Table 3.5-2 shows the command format.

Table 3.5-2 ID 1st Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type firstIDRequest(0)							
4	Acquirer ID (“ApplicationServiceProvider” format parameter)							
5								
6								
7								
8								
9								
10								
11								

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “firstIDRequest(0)” to indicate the ID 1st request command.

(4) Acquirer ID

This field indicates the Acquirer ID as “ApplicationServiceProvider” format parameter (refer to 3.5.3).

3.5.2.2.1.2 ID 1st Response Command

The ID 1st response command is used when the OBE notifies the roadside system of the OBE ID in response to the ID 1st request command sent from the roadside system. Table 3.5-3 shows the command format.

Table3.5-3 ID 1st Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type firstIDResponse(1)							
4	OBE ID("ObuID" format parameter)							
5								
6								
7								
8								
9								
10								
11								
12								

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier "operationCommand(1)" to indicate the normal command.

(3) Operation Type

This field is set to an identifier "firstIDResponse(1)" to indicate the ID 1st response.

(4) OBE ID

This field indicates the OBE ID as "ObuId" format parameter (refer to 3.5.3) to indicate the ID 1st response.

3.5.2.2.1.3 ID 2nd Request Command

The ID 2nd request command is used when the roadside system asks the OBE at the second time to give the OBE ID. Table 3.5-4 shows the command format.

Table 3.5-4 ID 2nd Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type secondIDRequest(2)							
4	Acquirer ID (“ApplicationServiceProvider” format parameter)							
5								
6								
7								
8								
9								
10								
11								

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “secondIDRequest(2)” to indicate the ID 2nd request.

(4) Acquirer ID

This field indicates the Acquuirer ID as “ApplicationServiceProvider” format parameter (refer to 3.5.3).

3.5.2.2.1.4 ID 2nd Response Command

The ID 2nd response command is used when the OBE notifies the roadside system of the OBE ID in response to the ID 2nd request command sent from the roadside system. Table 3.5-5 shows the command format.

Table 3.5-5 ID 2nd Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type secondIDResponse(3)							
4 :	ID 2nd Respence Information (“SecondIDResponse” format parameter)							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “secondIDResponse(3)” to indicate the ID 2nd response.

(4) ID 2nd Respence Information

This field indicates the ID 2nd response information as “SecondIDResponse” format parameter (refer to 3.5.3).

3.5.2.2.1.5 End Notice Command

The end notice command is used when the roadside system notifies the OBE that the ID acquisition processing is finished. Table 3.5-6 shows the command format.

Table 3.5-6 End Notice Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type endRequest(4)							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “endRequest(4)” to indicate the end notice command.

3.5.2.2.1.6 End Response Command

The end response command is used when the OBE gives response to the end notice command sent from the roadside system. Table 3.5-7 shows the command format.

Table 3.5-7 End Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type operationCommand(1)							
3	Operation Type endResponse(5)							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(3) Operation Type

This field is set to an identifier “endResponse(5)” to indicate the end response command.

3.5.2.2.2 OBU Denial Response Command

OBU denial response command is used when the OBE notifies the roadside system of negative acknowledgement. Table 3.5-8 shows the command format.

Table 3.5-8 OBU Denial Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type OBU denial response command obuDenialResponse(255)							
3	status							
4	data size of supplement information "supplementInfo"							
5 :	contents of supplement information "supplementInfo"							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier "obuDenialRespnose(255)" to indicate the negative acknowledge from OBE.

(3) status

This field is set to the reason of denial response. For details, refet to (Table 3.5-9).

(4) The length of supplement information

(a) This field is set to the length of successing supplement information. The unit is octet. When there are no supplement information (this case is default case), this filed is set to value "0".

(b) The contents of supplement information

This field is set to free information (the maximum length is 127 octets) as supplement information. When version number is not same, this field is set to own version ("versionIndex" parameter).

Table 3.5-9 Contents of "status" in OBU Denial Response Command in OBE ID Communication Application

No.	Meaning
0	Not used
1	Communication error
2	No OBE ID corresponding to specified acquirer ID
3	For future use
4	Version disagreed
5-10	For future use
11	Maintenance command failed
12	OBE ID not registered at all
13	OBE ID fully registered
14-31	For future use
32	Plain text send refused, authentication failed or no authentication
33-63	For in-application security
64-127	For future use
128-255	For private (Can be used arbitrarily by OBE)

3.5.2.2.3 Maintenance Commands

3.5.2.2.3.1 ID Registration Request Command

The ID registration request command is used when the roadside system asks the OBE to register the ID. Table 3.5-10 shows the command format.

Table 3.5-10 ID Registration Request Command

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type maintenanceCommand(2)							
3	Maintenance Type iDSetupRequest(0)							
4 :	OBE ID Resistration Information “ObuIDForRegistration” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “maintenanceCommand(2)” to indicate the maintenance command.

(3) Maintenance Type

This field is set to an identifier “iDSetupRequest(0)” to indicate the ID resistration request.

(4) OBE ID Resistration Information

This field is set to the OBE ID resistration information (“ObuIDForRegistration” format, Refer to 3.5.3).

3.5.2.2.3.2 ID Registration Response Command

The ID registration response command is used when the OBE notifies the roadside system that the ID registration request from the roadside system is completed. Table 3.5-11 shows the command format.

Table 3.5-11 ID Resistration Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type maintenanceCommand(2)							
3	Maintenance Type iDSetupResponse(1)							
4 :	OBE ID Resistration Information “ObuIDForRegistration” format parameter							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “maintenanceCommand(2)” to indicate the maintenance command.

(3) Maintenance Type

This field is set to an identifier “iDSetupResponse(1)” to indicate the ID resistration response.

(4) OBE ID Resistration Information

This field is set to the OBE ID resistration information (“ObuIDForRegistration” format, Refer to 3.5.3).

3.5.2.2.3.3 Registered ID Deletion Request Command

The registered ID deletion request command is used when the roadside system asks the OBE to delete a registered ID. Table 3.5-12 shows the command format.

Table 3.5-12 Registered ID Deletion Request Command

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type maintenanceCommand(2)							
3	Maintenance Type iDDeleteRequest(2)							
4	Acquirer ID("ApplicationServiceProvider" format parameter)							
5								
6								
7								
8								
9								
10								
11								

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier "maintenanceCommand(2)" to indicate the maintenance command.

(3) Maintenance Type

This field is set to an identifier "iDDeleteRequest(2)" to indicate the registered ID deletion request.

(4) Acquirer ID

This field is set to Acquirer ID for deleting OBE ID.

3.5.2.2.3.4 Registered ID Deletion Response Command

The registered ID deletion response command is used when the OBE notifies the roadside system that the registered ID deletion request from the roadside system is completed. Table 3.5-13 shows the command format.

Table 3.5-13 Registered ID Deletion Response Command

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type maintenanceCommand(2)							
3	Maintenance Type iDDeleteResponse(3)							
4	Acquirer ID("ApplicationServiceProvider" format parameter)							
5								
6								
7								
8								
9								
10								
11								

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier "maintenanceCommand(2)" to indicate the maintenance command.

(3) Maintenace Type

This field is set to an identifier "iDDeleteResponse(3)" to indicate the registered ID deletion response.

(4) Acquirer ID

This field is set to Acquirer ID for deleting OBE ID.

3.5.2.2.3.5 Registered ID List Request Command

The registered ID list request command is used when the roadside system asks the OBE to give the registered ID list. Table 3.5-14 shows the command format.

Table 3.5-14 Registered ID List Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type maintenanceCommand(2)							
3	Maintenance Type iDCheckRequest(4)							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “maintenanceCommand(2)” to indicate the maintenance command.

(3) Maintenace Type

This field is set to an identififier “iDCheckRequest(4)” to indicate the registered ID list request command.

3.5.2.2.3.6 Registered ID List Response Command

The registered ID list response command is used when the OBE notifies the roadside system of the registered ID list in response to the registered ID list request command from the roadside system. Table 3.5-15 shows the command format.

Table 3.5-15 Registered ID List ResponseCommand Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type maintenanceCommand(2)							
3	Maintenance Type iDCheckResponse(5)							
4	the number of Acqurir ID							
5 :	Acquirer ID List							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “maintenanceCommand(2)” to indicate the maintenance command.

(3) Maintenance Type

This field is set to an identififier “iDCheckResponse(5)” to indicate the registered ID list response command.

(4) APServiceProverList

(a) The number of Acquirer ID

This fiels is set to the number of acquirer ID in the registered acquirer ID list.

(b) Acquirer ID List

This field is set to the list of registered acquirer ID (“APServiceProviderList” format parameter).

3.5.2.2.3.7 ID Condition Change Request Command

The ID condition change request command is used when the roadside system asks the OBE to change the registered ID condition. Table 3.5-16 shows the command format.

Table 3.5-16 ID Condition Change Request Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type maintenanceCommand(2)							
3	Maintenace Type iDConditionChangeRequest(6)							
4-13	NewIDCondition							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “maintenanceCommand(2)” to indicate the maintenance command.

(3) Maintenace Type

This field is set to an identififier iDConditionChangeRequest(6)” to indicate the ID condition change request.

(4) NewIDCondition

This field is set to new condition of OBE ID (“NewIDConditon” format parameter, Refer to 3.5.3).

3.5.2.2.3.8 ID Condition Change Response Command

The ID condition change response command is used when the OBE notifies the roadside system that change is completed in response to the ID condition change request from the roadside system. Table 3.5-17 shows the command format.

Table 3.5-17 ID Condition Change Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	version				fill(0)			
2	Command Type maintenanceCommand(2)							
3	Maintenance Type iDConditionChangeResponse(7)							
4-13	NewIDCondition							

(1) version

This field indicates the application version.

(2) Command Type

This field is set to an identifier “maintenanceCommand(2)” to indicate the maintenance response command.

(3) Maintenance Type

This field is set to an identifier “iDConditionChangeResponse(7)” to indicate the ID condition change response.

(4) NewIDCondition

This field is set to the new condition of OBE ID (“NewIDConditon” format parameter, Refer to 3.5.3).

3.5.3 Definition of Parameter Types

```
ObuIDAcquisitionCommand::=SEQUENCE{
    versionIndex                Version,
    iDAcquisitionCommand        IDAcquisitionCommand
}

Version::= SEQUENCE {
    version     INTEGER(0..15),         -- value is set up to 1 at first
    fill        BIT STRING(SIZE(4))     -- value of encoding is set up to 0
}

IDAcquisitionCommand::=CHOICE{
    authenticateCommand         [0]     AuthenticateCommand,
                                            -- for securiy in the application
    operationCommand            [1]     OperationCommand,
    maintenanceCommand          [2]     MaintenanceCommand,
    dummy                       [3-254] NULL,  -- For future use
    obuDenialResponse           [255]   ObuDenialResponse
}

AuthenticateCommand::=CHOICE{
    authPath1   [0]     OCTET STRING,       -- for securiy in the application
    authPath2   [1]     OCTET STRING,       -- for securiy in the application
    authPath3   [2]     OCTET STRING,       -- for securiy in the application
    authPath4   [3]     OCTET STRING,       -- for securiy in the application
    dummy       [4-255] NULL                -- For future use
}

OperationCommand::=CHOICE{
    firstIDRequest      [0]     ApplicationServiceProvider,
                                        -- first ID request
    firstIDResponse     [1]     ObuID,
                                        -- response for the first ID request
```

```
        secondIDRequest     [2]     ApplicationServiceProvider,
                                        -- second ID request
        secondIDResponse    [3]     SecondIDResponse,
                                        -- response for the second ID request
        endRequest          [4]     NULL,  -- termination notification
        endResponse         [5]     NULL,
                                    -- response for the termination notification
        dummy               [6-255] NULL   -- For future use
}

maintenanceCommand ::=CHOICE{
        iDSetupRequest              [0]     ObuIDForRegistration,
                        -- request of ID registration
        iDSetupResponse             [1]     ObuIDForRegistration,
                        -- response for the request of ID registration
        iDDeleteRequest             [2]     ApplicationServiceProvider,
                        -- request of registered ID deletion
        iDDeleteResponse            [3]     ApplicationServiceProvider,
                        -- response for the request of registered ID deletion
        iDCheckRequest              [4]     NULL,
                        --request of registered ID list
        iDCheckResponse             [5]     APServiceProviderList,
                        -- response for the request of registered ID list
        iDConditionChangeRequest    [6]     NewIDCondition,
                        -- request of the ID condition change
        iDConditionChangeResponse [7]       NewIDCondition,
                        -- response for the request of the ID condition change
        dummy                       [8-255] NULL            -- For future use
}

ObuIDForRegistration::=SEQUENCE{
        applicationServiceProvider  ApplicationServiceProvider,
        iDCondition                 IDCondition,
        obuID                       ObuID
}
```

```
ApplicationServiceProvider::=OCTET STRING(SIZE(8))      -- acquier ID

IDCondition::=SEQUENCE{
    plaintextIDRefusal          BOOLEAN,
    -- whether ID transmits in plain text or not
    ciphertextIDRefusal         BOOLEAN,
    -- whether ID transmits in cipher text or not (for security in the application)
    mutualAuthentication        BOOLEAN,
    -- whether mutual authentication require or not (for security in the application)
    userApproval                BOOLEAN,
    -- whether operation of transmit permission require or not
    idUnlock                    BOOLEAN,        -- whether OBE ID delete or not
    spf                         BOOLEAN,
    -- whether common SPF security transmit or not
    fill                        BIT STRING(SIZE(10))    -- For future use
}

ObuID::=SEQUENCE{
    fill                        BIT STRING(SIZE(7)),
    originalObuID               OCTET STRING(SIZE(8)), -- OBE ID
    mACForOriginalText                  MACForOriginalText          OPTIONAL
}

MACForOriginalText::=SEQUENCE{
    encryptionAlgorithmId       INTEGER(0..255),
        -- encryption algorithm for MAC (for security in the application)
    keyNumber                   INTEGER(0..255),
        -- key number for MAC (for security in the application)
    MAC                         OCTET STRING(SIZE(4))
        -- for security in the application
}

ObuDenialResponse ::=SEQUENCE{
    status                  INTEGER(0..255),
```

```
    supplementInfo      OCTET STRING(SIZE(0..255)) -- supplement information
}

SecondIDResponse::=SEQUENCE{
    encryptionAlgorithmId       INTEGER(0..255),
        -- encryption algorithm for ID secret (for security in the application)
    keyNumber                   INTEGER(0..255),
        -- key number for ID secret (for security in the application)
    encryptedId                 OCTET STRING
        -- encryption ID information (for security in the application)
}

APServiceProviderList ::=SEQUENCE(0..255) OF ApplicationServiceProvider

NewIDCondition ::=SEQUENCE{
    applicationServiceProvider          ApplicationServiceProvider,
    iDCondition                         IDCondition
}
```

3.5.4 Relationship with Other Standards

- Use AID=18 (DSRC APPLICATION SUB LAYER)
- Use Local Port Control Protocol in NCP of ASL.
- Use 0x0C00 in port number.
- Use Unidirectional data transmit transaction service in two transaction service provided by LPCP.

3.5.5 Communication Procedures

3.5.5.1 ID Acquisition Procedure

The communication procedure for acquiring the OBE ID is as follows:

(1) The roadside system sends “firstIDRequest” to the OBE.

(2) When receiving “firstIDRequest”, the OBE refers to the ID registration information, and sends “firstIDResponse” or “ObuDenialResponse” to the roadside system as follows:

(a) When the condition of the OBE ID corresponding to the specified acquirer ID is “plaintextIDRefusal (false (0))", the OBE sends “firstIDResponse” to the roadside system for notifying the OBE ID.When the condition is “plaintextIDRefusal (true (1))", the OBE sends “obuDenialResponse” whose status is “32: Plain text send refused, authentication failed or no authentication” to the roadside system.

(b) When the OBE ID is not registered at all in the OBE, the OBE sends “ObuDenialResponse” whose status is “12: OBE ID not registered at all” to the roadside system.

(c) When the OBE ID corresponding to the specified acquirer is not registered, the OBE sends “obuDenialResponse” whose status is “2: No OBE ID corresponding to specified acquirer ID” to the roadside system.

The communication procedure for terminating the OBE ID acquisition processing is as follows:

(1) The roadside system sends “endRequest” to the OBE.

(2) When receiving “endRequest”, the OBE finishes the OBE ID acquisition processing, and sends “endResponse” to the roadside system.

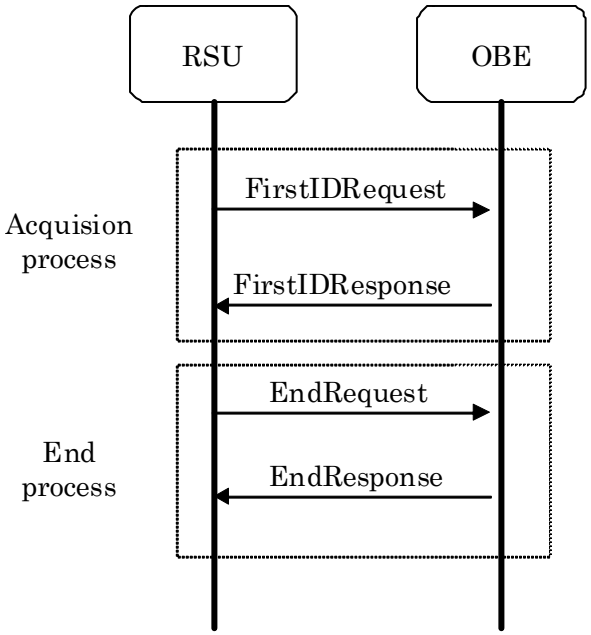


Figure 3.5-1 Example sequence of OBE ID acquisition processing

Reference:The figure below shows a sequence example when the corresponding ID condition is “plaintextIDRefusal (true (1))”, “ciphertextIDRefusal (false (0))” and “mutualAuthentication (true (1)” though this case is outside the range of specification described in this document.

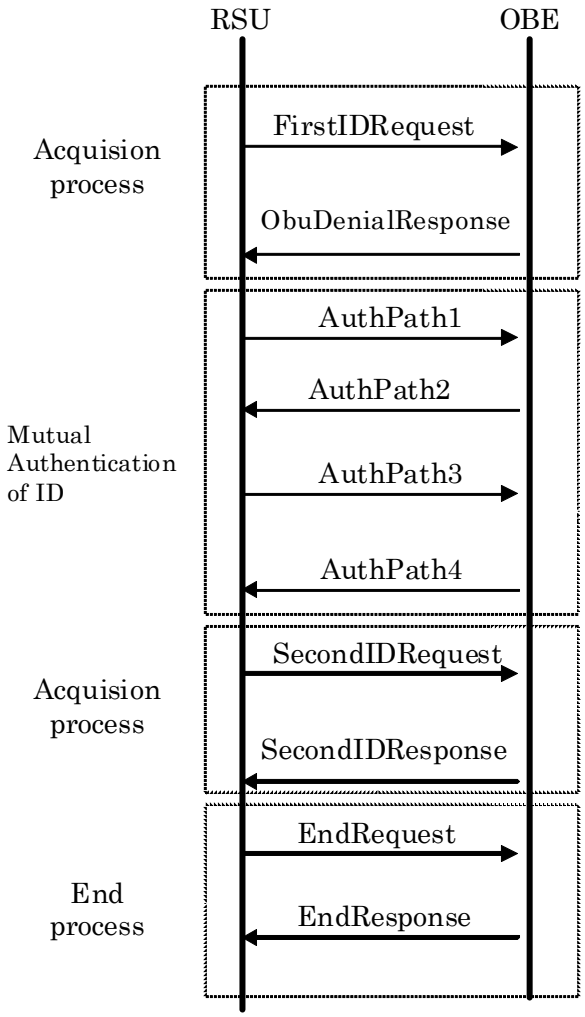


Figure 3.5-2 Example sequence of OBE ID acquisition processing with security function of OBE ID communication application

3.5.5.2 ID Maintenance Sequence

This paragraph describes communication procedures for maintenance of the OBE ID.

3.5.5.2.1 Acquisition of registered ID list

(1) The roadside system sends “IDCheckRequest” to the OBE.

(2) When receiving "IDCheckRequest", the OBE sends the acquirer ID list registered in the OBE “APServiceProviderList” as “IDCheckResponse” to the roadside system.

3.5.5.2.2 Change of ID condition

(1) The roadside system sends “IDConditionChangeRequest” to the OBE.

(2) When receiving "IDConditionChangeRequest", the OBE refers to the contents of its “NewIDCondition”, and sends “IDConditionChangeResponse” or “ObuDenialResponse” to the roadside system as follows:

(a) The OBE updates the OBE ID information having the specified acquirer ID, and sends “IDConditionChangeResponse” to the roadside system for notifying that change is completed.

(b) When there is no OBE ID corresponding to the specified acquirer ID, the OBE sends “ObuDenialResponse” whose status is “2: No OBE ID corresponding to specified acquirer ID” to the roadside system.

(c) When the OBE ID is not registered at all, the OBE sends “ObuDenialResponse” whose status is “12: OBE ID not registered at all” to the roadside system.

3.5.5.2.2.1 Deletion of registered ID

(1) The roadside system sends to the OBE “IDDeleteRequest” containing the acquirer ID corresponding to the OBE ID to be deleted.

(2) When receiving "IDDeleteRequest", the OBE refers to the acquirer ID contained in “IDDeleteRequest”, and sends “IDDeleteResponse” or “ObuDenialResponse” to the roadside system as follows:

(a) When there is OBE ID corresponding to the specified acquirer ID whose condition is “idUnlock (true (1)) enabling deletion”, the OBE deletes the OBE ID, and sends “IDDeleteResponse” to the roadside system. If the OBE ID condition is “idUnlock (false (0)) disabling deletion”, the OBE sends “ObuDenialResponse” whose status is “11: Maintenance command failed” to the roadside system.

(b) When there is no OBE ID corresponding to the specified acquirer ID, the OBE sends

“ObuDenialResponse” whose status is “2: No OBE ID corresponding to specified acquirer ID” to the roadside system.

(c) When the OBE ID is not registered at all, the OBE sends “ObuDenialResponse” whose status is “12: OBE ID not registered at all” to the roadside system.

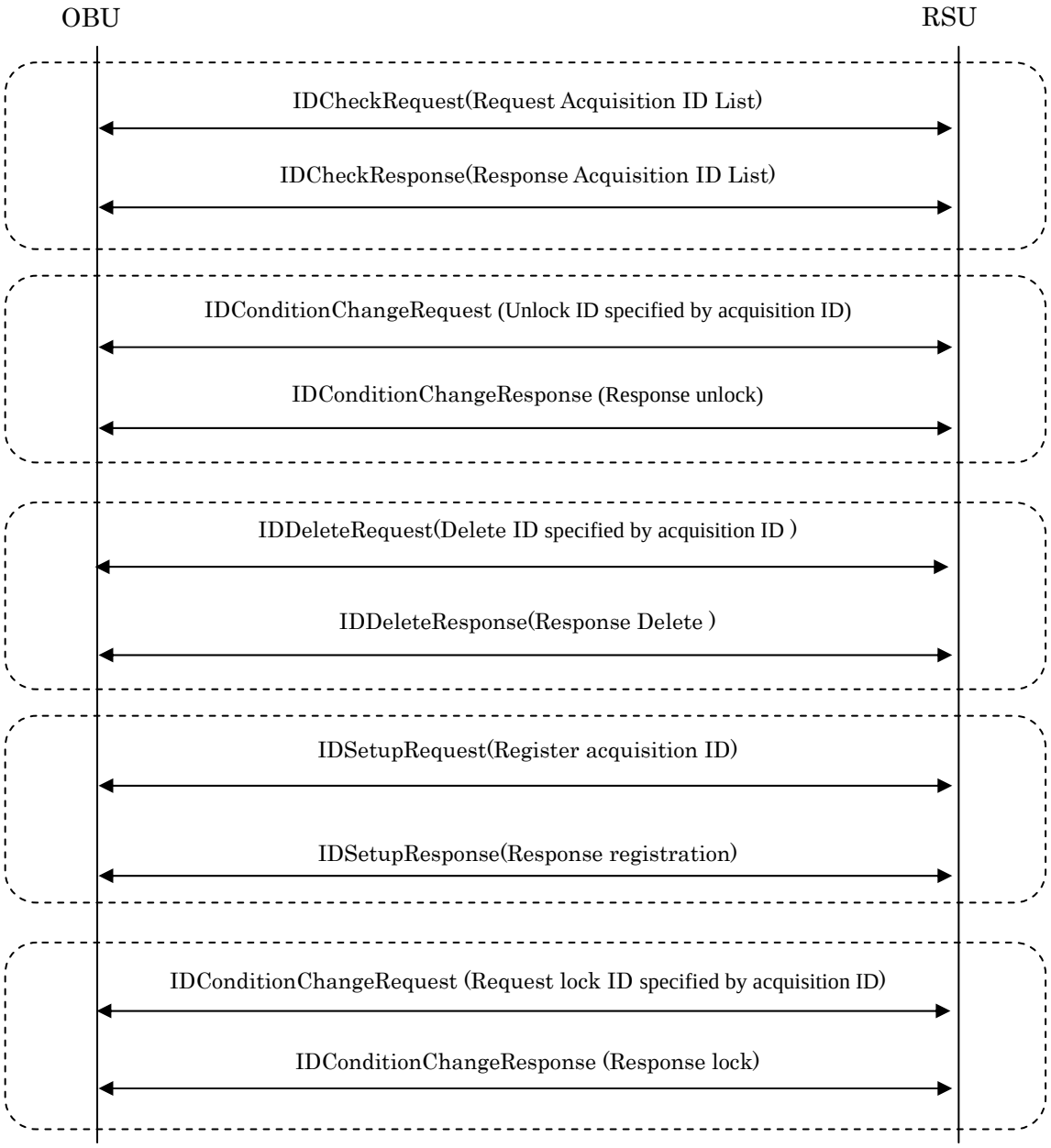
3.5.5.2.2.2 Registration of ID

(1) The roadside system sends “IDSetupRequest” to the OBE.

(2) When receiving “IDSetupRequest”, the OBE sends “IDSetupResponse” or “ObuDenialResponse” to the roadside system as follows:

(a) When there is a registration area for new OBE ID, the OBE registers the OBE ID in accordance with the OBE registration information specified in “IDSetupRequest”, and sends “IDSetupResponse” to the roadside system for notifying that registration is completed.

(b) When the maximum allowable number of registered OBE IDs is reached and there is no registration area for new OBE ID, the OBE sends “ObuDenialResponse” whose status is “13: OBE ID fully registered" to the roadside system.



Note:It can be executed independently in units of area surrounded by broken lines.

Figure 3.5-3 Example of maintenance sequence

3.6 OBE Basic Indication Application

3.6.1 Function Overview

The OBE basic indication application notifies the basic instruction information provided fee information from the external server connected to the roadside equipment.

3.6.2 Command

3.6.2.1 Command System

Commands used in the OBE basic instruction application consist of the normal command and the denial response command from the OBE. And the normal commands consist of “basic instruction notice command from the roadside system to the OBE” and “basic instruction response command from the roadside system to the OBE”.

3.6.2.2 Command Format

3.6.2.2.1 Normal Commands

3.6.2.2.1.1 Basic Instruction Notice Command

The basic instruction notice command is used when the roadside system notifies the OBE of the basic instruction information. Table 3.6-1 shows the command format.

Table 3.6-1 Basic Instruction Notice Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Command Type operationCommand(1)							
2	Operation Type bOIRequest(0)							
3-36	OBE Basic Instruction Information “BasicObuIndication” format parameter							

(1) Command Type

This field is set to an identifier “operationCommand(1)” to indicate the normal command.

(2) Operation Type

This field is set to an identifier “bOIRequest(0)” to indicate the notice command of basic instruction.

(3) OBE Basic Instruction Information

This field is set to the OBE basic instruction information ("BasicObuIndication" format, Refer to 3.6.3).

(a) versionIndex

This field indicates the version of OBE basic instruction application. The current value is 1.

(b) transactionResult

It indicates the communication result with the roadside system. Three values shown in Table 3.6-2 are defined, and other values are reserved for future use.

Table 3.6-2 Communication Result Format

Value	Meaning
0	The service is normally terminated without charge.
64	The service is abnormally terminated.
128	The service is normally terminated with charge.

(c) supplement

This field indicates the supplement information of communication result from road side unit. It is used 8-bit code of JIS X 0201. "0x00 00 00 00 00" is given when there is no supplement information.

(d) time

It is used to notify the time. Table 3.6-3 shows the format. "0x00 00 00 00" is given when there is no effective time information.

Table 3.6-3 Time Format

Stored order	Item	Bit size	Data type
1	year	7	INTEGER(0..127)
2	month	4	INTEGER(0..12)
3	day	5	INTEGER(0..31)
4	hour	5	INTEGER(0..23)

5	minute	6	INTEGER(0..59)
6	second	5	INTEGER(0..29)

Note: The year is expressed as relative year from 1997. And, the second indicates in 2 seconds unit.

(e) amount

It is used to notify the fee/toll. Table 3.6-4 shows the format.

Table3.6-4 Fee format

Stored order	item	Bit size	Data type
1	fee	24	INTEGER(-8,388,608..8,388,607)
2	unit	16	BCD(4)

Note: "0x0392" specified in ISO4217 is stored as the unit.

3.6.2.2.1.2 Basic Instruction Response Command

The basic instruction response command is used when the OBE notifies the roadside system of the normal operation in response to the basic instruction response command. Table 3.6-5 shows the command format.

Table 3.6-5 Basic Instruction Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Command Type operationCommand(1)							
2	Operation Type bOIResponse(1)							

(1) Command Type

This field is set to an identifier "operationCommand(1)" to indicate the normal command.

(2) Operation Type

This field is set to an identifier "bOIResponse(1)" to indicate the response command of basic instruction.

3.6.2.2.2 OBE Denial Response Command

The OBE denial response command is used when the OBE notifies the roadside system of negative acknowledgement in response to the basic instruction notice command. Table 3.6-6 shows the command format.

Table 3.6-6 OBE Denial Response Command Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Command Type obuDenailResponse(255)							
2	Status							
3	The length of supplement information							
4 :	The contents of supplement information							

(1) Command Type

This field is set to an identifier “obuDenialResponse(255)” to indicate the normal command.

(2) Status

This field is set to the reason of denial response. For details, refer to Table 3.6-7.

Table 3.6-7 Status Code

Status code	Description
0	Not use
1	Communication error (instruction information and OBE inside error)
2-3	For future use
4	The version incompatible
5-127	For future use
128-255	For private (note)

Note: OBE can use these code arbitrarily.

(3) Supplement Information

(a) The length of supplement information

This field is set to the length of successing supplement information. The unit is

octet. When there are no supplement information (this case is default case), this filed is set to value “0”.

(b) The contents of supplement information

This field is set to free information (the maximum length is 127 octets) as supplement information. When version number is not same, this field is set to own version (“versionIndex” parameter).

3.6.3 Definition of Parameter Types

```
BasicObuIndicationCommand ::=CHOICE{
    dummy                [0]      NULL,
    operationCommand     [1]      OperationCommand,
    dummy                [2-254]  NULL,          -- For future use
    obuDenialResponse    [255]    ObuDenialResponse
}

OperationCommand::=CHOICE{
    bOIRequest           [0]      BasicObuIndication,--basic instruction notice
    bOIResponse          [1]      NULL,          --basic instruction response
    dummy                [2-255]   NULL          --For future use
}

ObuDenialResponse ::=SEQUENCE{
    status               INTEGER(0..255),
    supplementInfo       OCTET STRING(SIZE(0..255)) --supplement information
}
BasicObuIndication ::=SEQUENCE{
    versionIndex         INTEGER(0..255),
    transactionResult    INTEGER(0..255),
    supplement           OCTET STRING(SIZE(5)),
    dummy1               OCTET STRING(SIZE(12)),
    time                 OCTET STRING(SIZE(4)),
    dummy2               OCTET STRING(SIZE(1)),
    amount               OCTET STRING(SIZE(5)),
    dummy3               OCTET STRING(SIZE(5))
}
```

3.6.4 Relationship of Other Standard

- Use AID=18 (DSRC APPLICATION SUB LAYER)
- Use Local Port Control Protocol in NCP of ASL.
- Use 0x0C08 in port number of LPCP.
- Use Unidirectional data transmit transaction service in two transaction service provided by LPCP.

3.6.5 Communication Procedure

This subsection describes the communication procedure of the OBE basic instruction application.

(1) The roadside system notifies the on-board equipment of the OBE basic instruction information using the basic instruction notice command.

(2) When the OBE receives the basic instruction notice command, it refers to the OBE basic instruction information, and outputs the contents. When the output is completed, the OBE sends the basic instruction response command to the roadside system.

(3) When the OBE rejects the OBE basic instruction command due to the contents of the OBE basic instruction or the OBE status in the step (2), it sends the OBE denial response command to the roadside system.

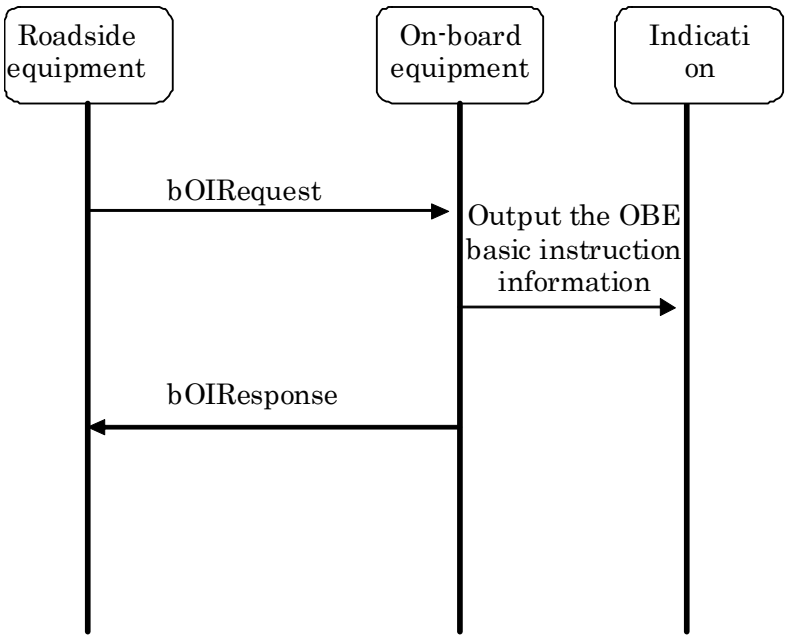


Figure 3.6-1 sequence example of OBE basic instruction (normal operation)

[BLANK]

Annex A Relationship between DSRC Services and Basic Application Interface

A.1 Postulated Examples of DSRC Services and Required Functions

The table below shows provision places, concrete contents and required functions of postulated DSRC services.

Table A1-1 Functions required in DSRC services

Postulated DSRC service examples			Required function	
Provision place		Concrete contents	Charge settlement	Information exchange
(1)	Gas station	Fuelling, and sales and purchasing of goods, etc.	X	
		Information distribution, sales and purchasing of maps, music, etc.	X	X
		Guidance of services (such as Vehicle washing and oil change)		X
		Services for members such as loyalty Point Card		X
(2)	Vehicle dealer	Control of customer, and control and provision of maintenance history		X
		Sales and purchasing of goods, etc.	X	
		Information distribution, sales and purchasing of maps, music, etc.	X	X
		Update, sales and purchasing of software for OBE (such as navigator)	X	X
(3)	Parking lot	Control of entrance and exit (for monthly contract)		X
		Guidance of parking position (on pay-by-the-hour basis)		X
		Charge and payment of parking fee	X	
(4)	Fast food restaurant, convenience store, etc	Information distribution, sales and purchasing of various goods, discount goods, etc.	X	X
		Reservation, sales and purchasing of various tickets	X	X
		Information distribution, sales and purchasing of maps, music, etc.	X	X

(5)	Ferry landing	Reservation and boarding procedure		X
		Charge and payment of ferry fee	X	
		Guidance of parking position		X
(6)	Store/facility notice and guidance	Various advertisements and guidance of store/facility		X
		Reservation for use of store/facility		X
(7)	Expressway service area and parking area	Provision of traffic information, restriction information, road information, etc.		X
		Provision of services, facility guidance, sightseeing information, etc.		X

A.2 Detailed Function Analysis

A.2.1 Requirements in system realizing charge settlement

For realizing the charge settlement function in the DSRC system, the following two methods are postulated:

- Settlement method in which the identification information held in the OBE is linked with the settlement means
- Settlement method in which the IC Card mounted in the OBE is accessed

(1) Settlement Processing using OBE ID

The following functions are required to link the identification information held by the OBE with the settlement means, and then settle the charge. The infrastructure should have not only the following functions but also the function to realize the settlement corresponding to the information (ID) specific to the OBE.

(a) Function to give various instructions to the OBE (Function equivalent to or expanded from the function of the ETC system)

(b) Function to access the information (ID) specific to the OBE

(c) Function to access the memory incorporated in the OBE (for accumulating the use information, etc.)

(2) Card settlement system

The following functions are required to access the IC Card mounted in the OBE, and then settle the charge using the IC Card. The infrastructure should have not only the following functions but also the settlement function specific to the IC Card.

(a) Function to give various instructions to the OBE (Function equivalent to or expanded from the function of the ETC system)

(b) Function to access the IC Card mounted in the OBE

(c) Function to access the memory incorporated in the OBE (for accumulating the use information, etc.)

A.2.2 Requirements for system realizing information exchange

For realizing the information exchange function in the DSRC system, the following two methods are postulated:

- Exchange method in which the counterpart is authenticated using the identification information held by the OBE, and then information is exchanged between the counterpart and the memory incorporated in the OBE

- Exchange method in which the OBE is used as communication means, and then information is exchanged between the information terminal unit connected to the OBE and the roadside system

(1) OBE specific information control system

The following functions are required to authenticate the counterpart using the identification information held by the OBE, and then exchange information between the counterpart and the memory incorporated in the OBE.

The infrastructure should have not only the following functions but also the information processing function.

(a) Function to give various instructions to the OBE (Function equivalent to or expanded from the function of the ETC system)

(b) Function to access the information (ID) specific to the OBE

(c) Function to access the memory incorporated in the OBE (for accumulating the use information, etc.)

(2) Information provision system interlocking with the image display unit

The following functions are required to use the OBE as communication means, and then exchange information between the information terminal unit connected to the OBE and the roadside system.

The infrastructure and information terminal unit connected to the OBE should have not only the following functions but also the information processing function.

(a) Function to connect the information server through IP connection, and access information in accordance with requests from the OBE (Request/response type information provision)

(b) Function to distribute the URL of the start page for provided services and diversified

information from the information server to the OBE (Push type information provision)

A.2.3 Correspondence between required functions and processing flows

The figure shows the correspondence between the required functions extracted in the previous paragraph and the processing flow in each service scene.

A.2.3.1 Charge settlement

(1) Linking settlement system

System to perform settlement using the personal information linked with the specific information held by the OBE

(a) Function to give various instructions to the OBE (Function equivalent to or expanded from the function of the ETC system)

(b) Function to access the information (ID) specific to the OBE

(c) Function to access the memory incorporated in the OBE (for accumulating the use information, etc.)

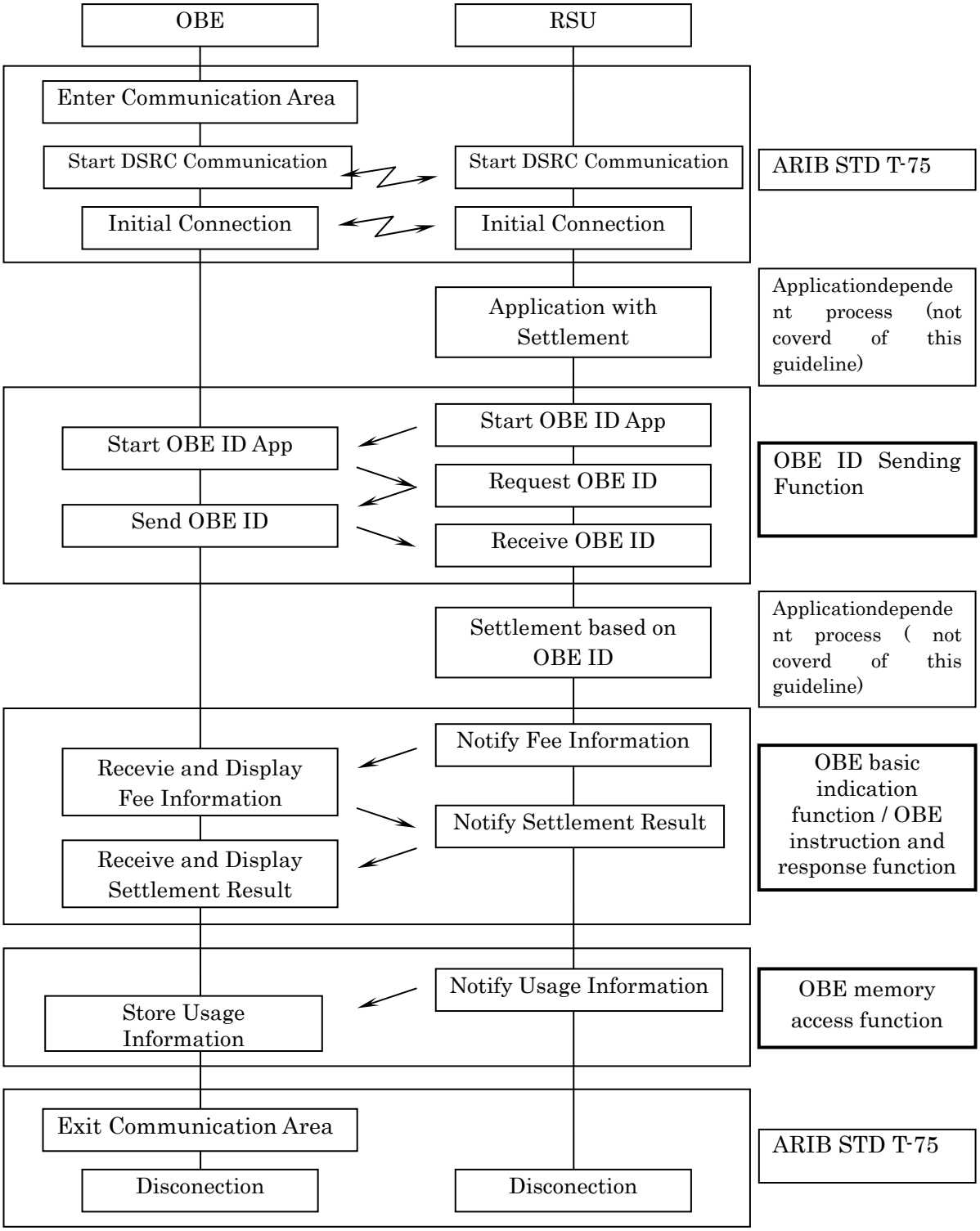


Figure A2-1 Procedure of Linking settlement system

(2) Card settlement system

System to directly perform settlement using a contact type IC Card (such as Credit Card and Prepaid Card)

(a) Function to give various instructions to the OBE (Function equivalent to or expanded from the function of the ETC system)

(b) Function to access the IC Card mounted in the OBE

(c) Function to access the memory incorporated in the OBE (for accumulating the use information, etc.)

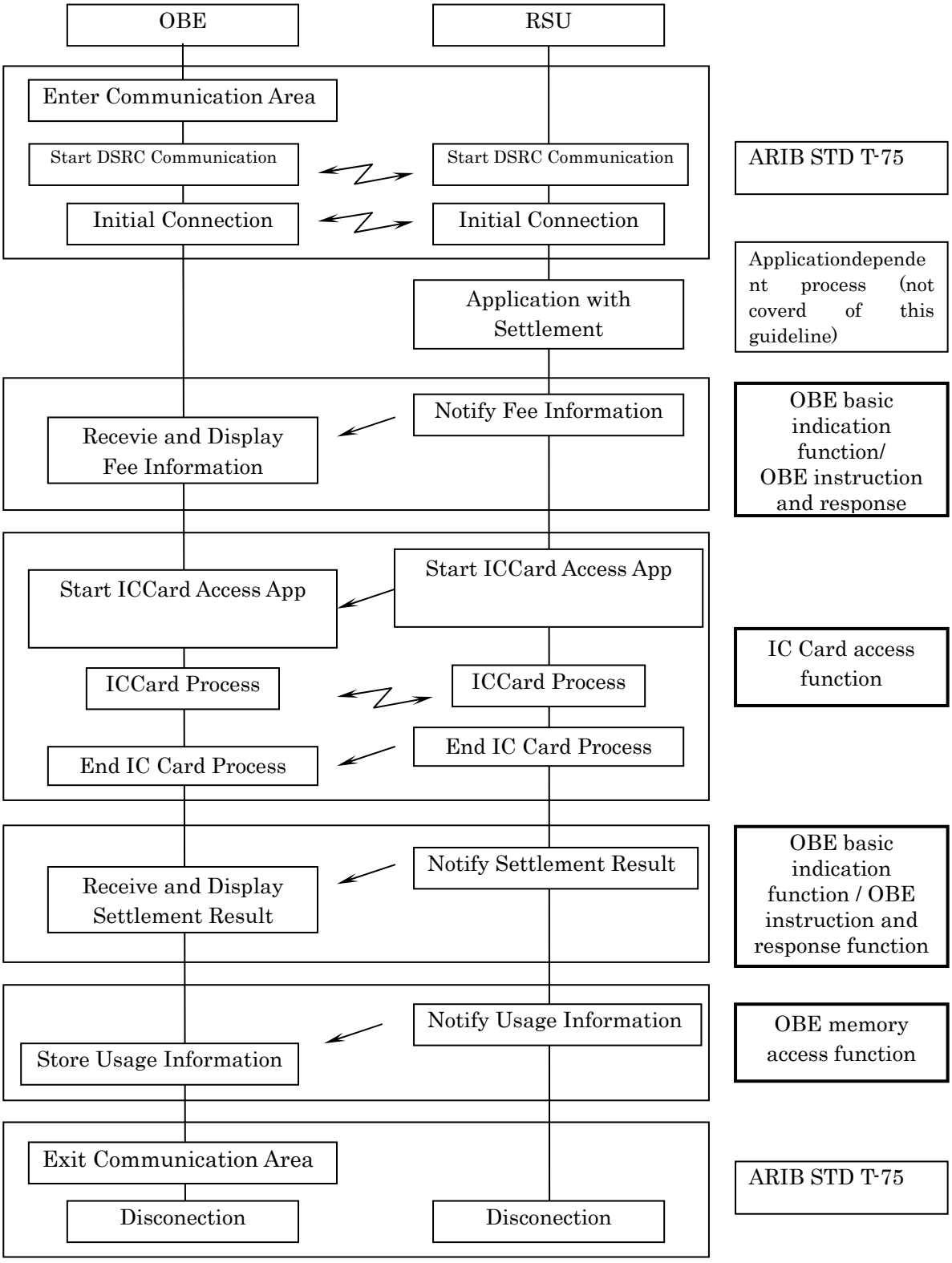


Figure A2-2 Procedure of Card settlement system

A.2.3.2 Information exchange

(1) Vehicle specific information control system

System to control the information specific to the Vehicle for customer control by Vehicle dealers, entrance/exit control in monthly-paid parking lot, etc.

(a) Function to give various instructions to the OBE (Function equivalent to or expanded from the function of the ETC system)

(b) Function to access the information (ID) specific to the OBE

(c) Function to access the memory incorporated in the OBE (for accumulating the use information, etc.)

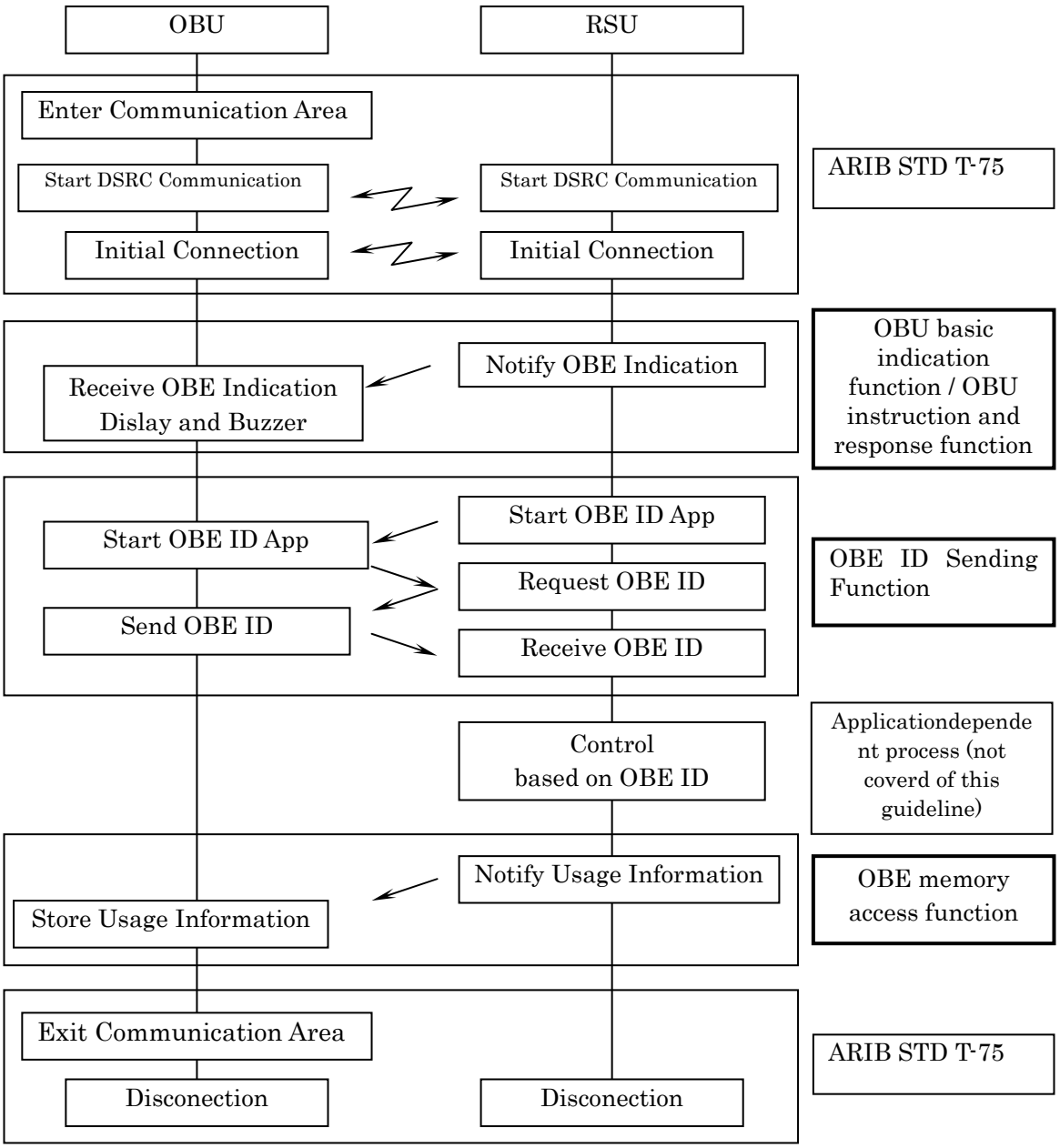


Figure A2-3 Procedure of Vehicle specific information control system

(2) Information provision system interlocking with the image display unit

System to provide traffic information and road information, and distribute information for maps, music, etc. through interlock with the information display unit (such as Vehicle navigation equipment) attached to the OBE

(a) Function to connect the information server through IP connection, and access information in accordance with requests from the OBE (Request/response type information provision)

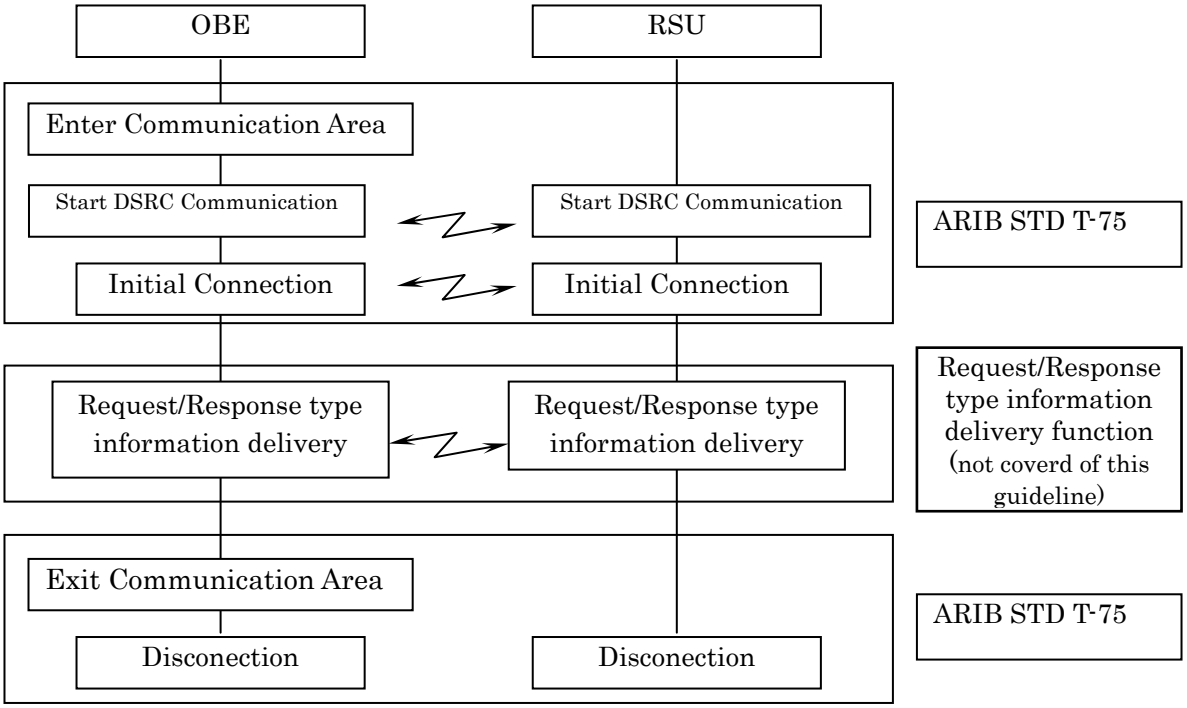


Figure A2-4 Processing flow in the request/response type information provision system

(b) Function to distribute the URL of the start page for provided services and diversified information from the information server to the OBE (Push type information provision)

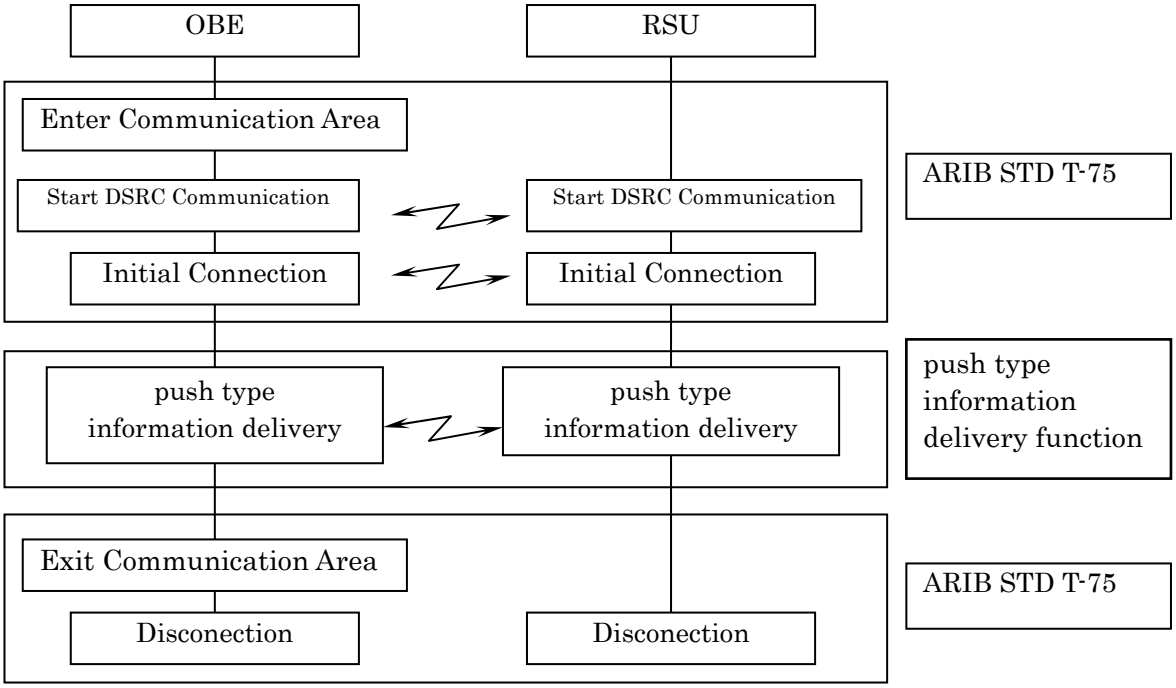


Figure A2-5 Processing flow in the push type information delivery system

Annex B Relationship with Security Platform

This section describes the relationship with the security platform located between the DSRC basic application interface and the local port protocol (LPP).

The security platform (DSRC-SPF) executes mutual authentication between the DSRC OBE and the roadside system, and authenticates the equipment. The DSRC-SPF is available for cryptographic communication of the DSRC basic application interface by using a key exchanged in mutual authentication. The security used by the DSRC-SPF is selectable among several types.

B.1 Security Platform Configuration

The security platform located between the DSRC basic application interface and the LPP as shown in Attached fig. B1-1 executes the security type negotiation, mutual authentication and key exchange processing using the local port number (LP1) assigned to the DSRC-SPF, as well as encrypts send data given by the basic application interface and decrypts received data given by the LPP using the selected security type and exchanged key acquired in the authentication/key exchange phase.

Each basic application interface has two ports, a port using the DSRC-SPF (secure port LP3) and a port not using the DSRC-SPF (normal port LP2), and incorporates the function to select whether data is processed using the DSRC-SPF in units of transaction of the LPP (through sending to the secure port) or the DSRC-SPF is bypassed (through sending to the normal port).

Note:It is not necessary to increase the local port for each security type even if security type choices available in the SPF increase.

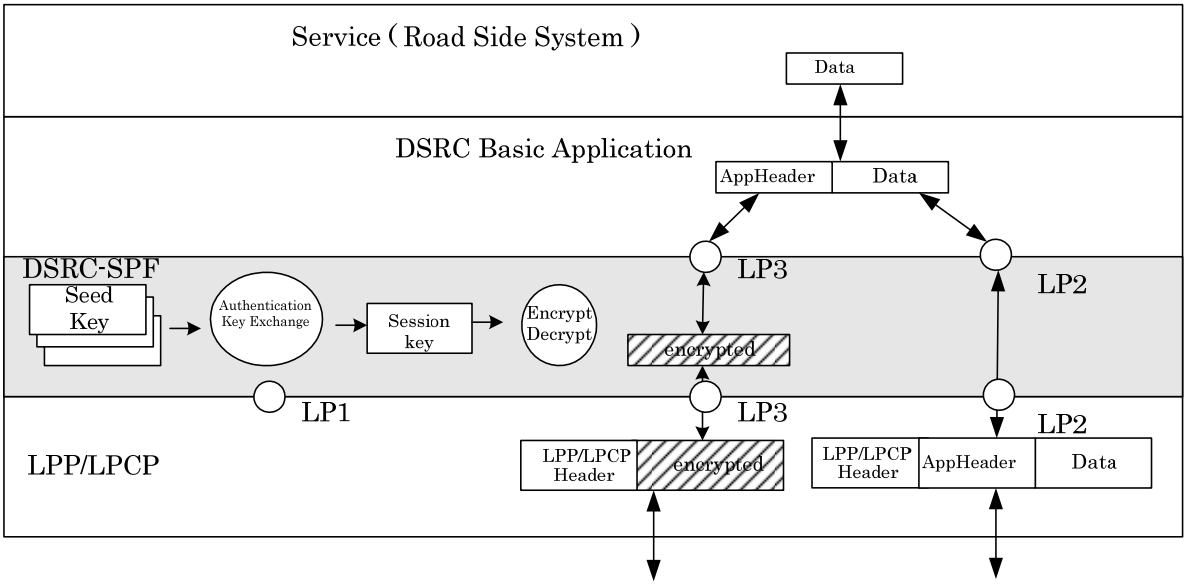


Figure B1-1 Security Platform configuration

Details of the security processing specified by the security type are outside the range of this guideline, and shall be specified separately.

B.2 Local Port Number List

When the SPF is not used, the conventional area from “0x0C00” to “0x0C1F" is used as the normal port area for the local port number of the basic application interface, and the area from “0X0C20” to “0x0C3F” is used as the secure port area. The classification and layout of the basic application interface in the secure port area conforms to the classification and layout of the basic application interface in the normal port area.

The port number “0x0C3F” is used as the control port for the basic application interface security platform, and used in the authentication/key exchange phase.

Table B2-1 shows the list of local port numbers of the basic application interface.

TableB2-1 Local Port Number of Basic Application

Normal Port	Secure Port	Application	Remarks
0x0C00	0x0C20	OBE ID Communication Application	<data flow> RSU<-OBU
0x0C01-0x0C07	0x0C21-0x0C27	For future use	
0x0C08	0x0C28	OBE Basic Indication Application	<data flow> RSU->OBU
0x0C09	0x0C29	OBE Instruction Response Application	
0x0C0A	0x0C2A	Push-type information delivery application	
0x0C0B-0x0C0F	0x0C2B-0x0C2F	For future use	
0x0C10	0x0C30	ICCard Access Application	<data flow> RSU<->OBU using IC Card
0x0C11	0x0C31	Contactless IC Smart Card Access Application	
0x0C12-0x0C17	0x0C32-0x0C37	For future use	
0x0C18	0x0C38	OBE Memory Access Application	<data flow> RSU<->OBU using Memory
0x0C19-0x0C1F	0x0C39-0x0C3E	For future use	
-	0x0C3F	Security Platform Management Entity	

B.3 Important Notice about Basic Application Interface using SPF

B.3.1 Relationship with in-application security

The in-application security attribute and common SPF are available independently. Accordingly, it is possible in secure ports to use the SPF independently or use both the SPF and the in-application security together.

When using only the common SPF, use data without any security (as plain text) in the application, and execute encryption and decryption in the SPF.

B.3.2 Access control using SPF for OBE

It is postulated that communication is enabled at secure ports in each application only after the SPF authentication/key exchange phase is completed.

(1) OBE ID communication application

For enabling sending the OBE ID using the SPF in the OBE ID communication application, set the OBE ID to "spf (true (1))".

When using the SPF to control accesses of maintenance commands, disable maintenance commands from normal ports, and enable maintenance commands only from secure ports (except in the mutual confirmation test, etc.).

(2) IC card access application

Accesses from normal ports are disabled in the IC card access application (except in the mutual confirmation test, etc.).

(3) Push type information delivery application

Issue the client information notice "ClientInformation" when secure ports become ready for communication in the push type information delivery application.

When normal ports are used at the same time, it is possible to issue "ClientInformation" for each port and give different settings of receivable contents to secure ports and normal ports.

(4) OBE memory access application

It is recommended to use secure port when using commands with password in the OBE memory access application.

Annex C Service Application Examples using Basic Application Interface

This section describes service application examples using the basic application interface specification.

This section describes the already introduced DSRC system in parking lots, demonstration experiment system in parking lots and demonstration experiment system for information providing services on access roads of expressway service areas, etc. This section is available as reference for utilizing the DSRC basic application interface.

C.1 Settlement Processing using OBE ID

The figure below shows a transaction example in the settlement processing system using the OBE ID application interface.

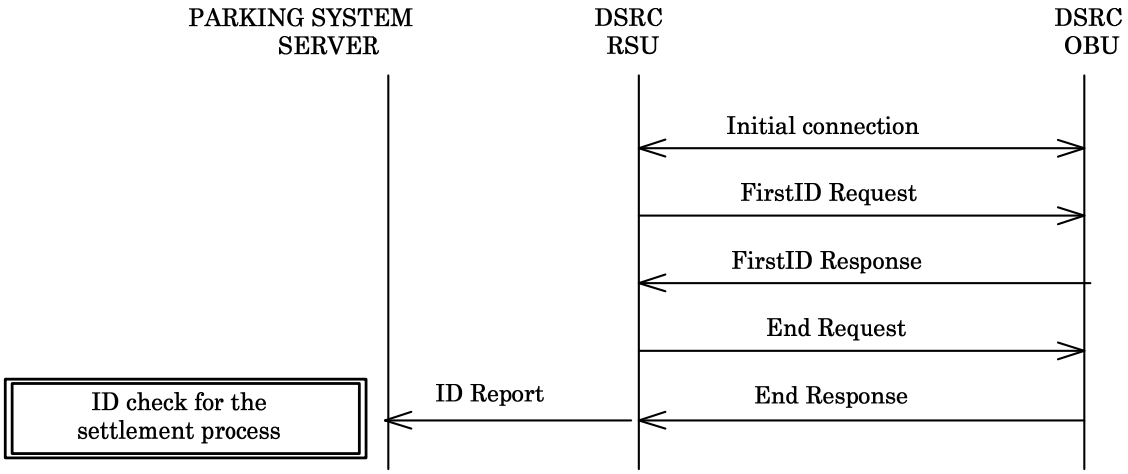


Figure C1-1 Communication transaction example (for plain text) in the settlement processing system using the OBE ID application interface

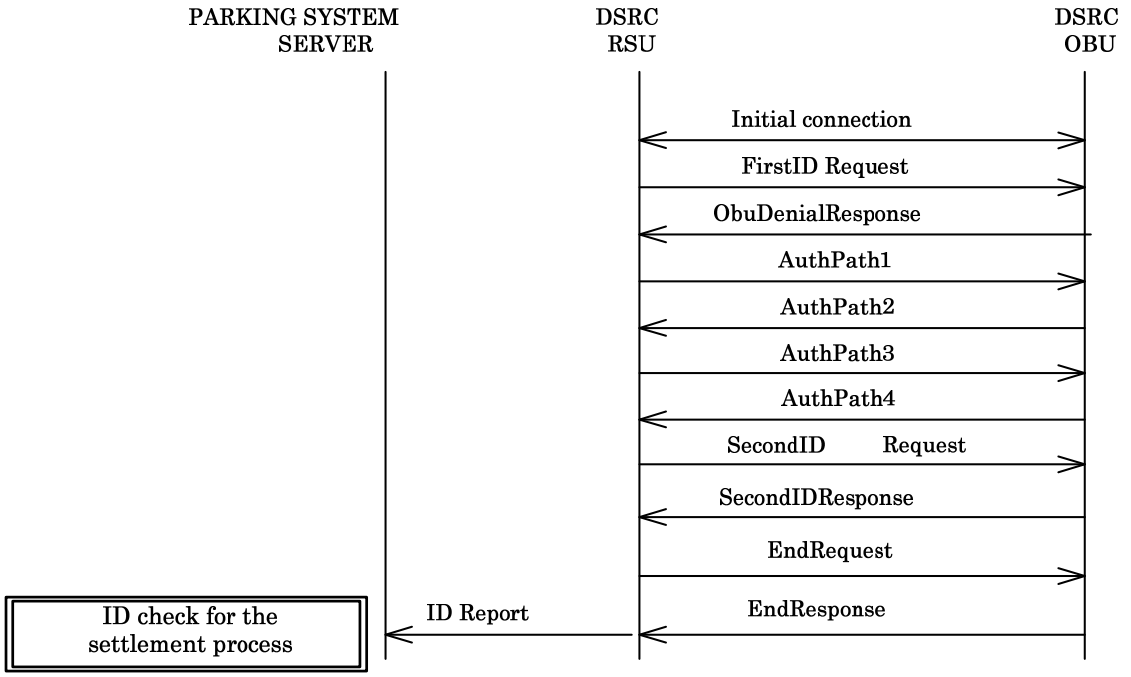


Figure C1-2 Communication transaction example (for encrypted text) in the settlement processing system using the OBE ID application interface

C.2 Prepaid Type Settlement Processing using IC Card Access

The figure below shows a transaction example of the prepaid type settlement processing system using both the IC card access application interface and the push type information delivery application interface.

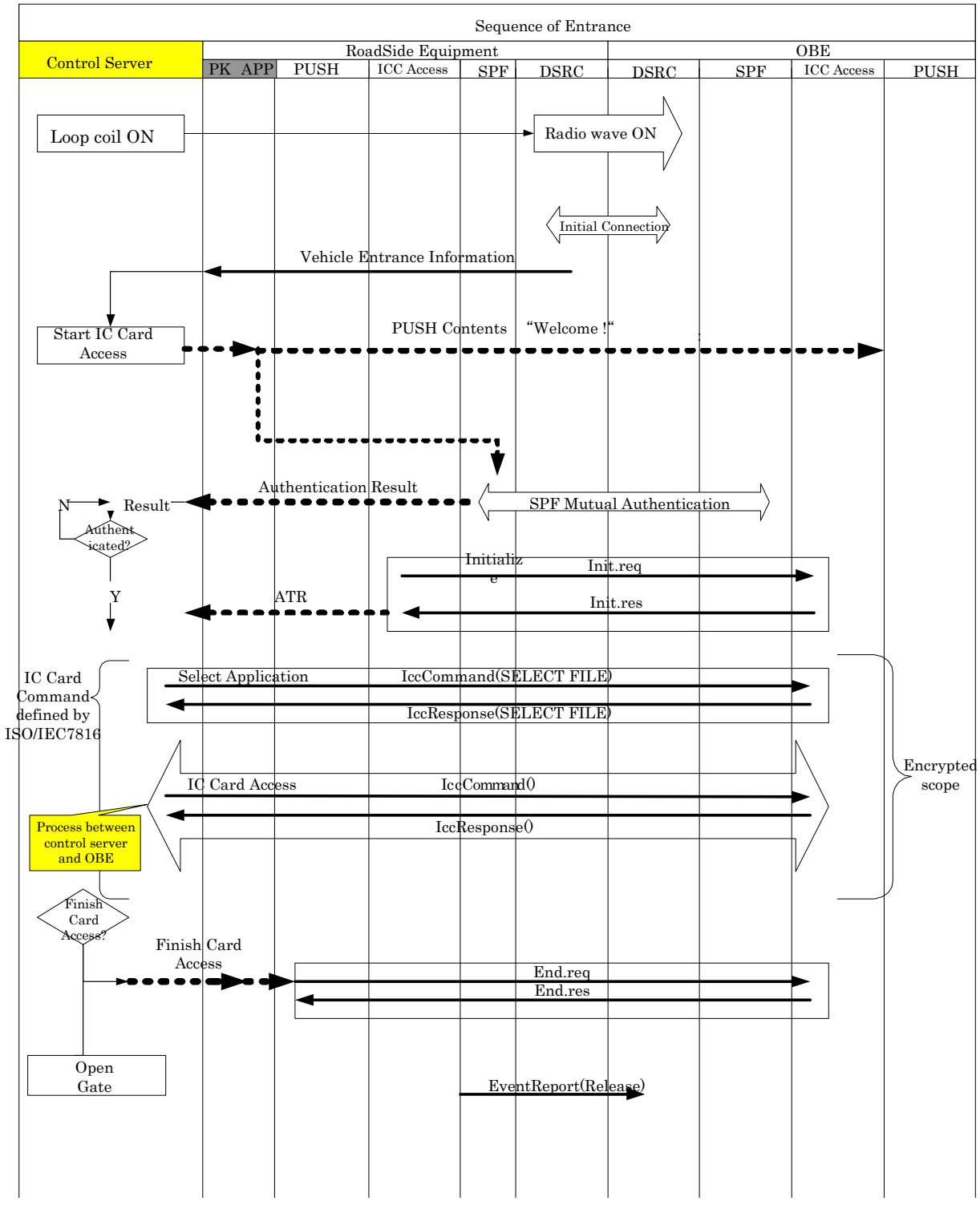


Figure C2-1 Processing Sequence of Entrance

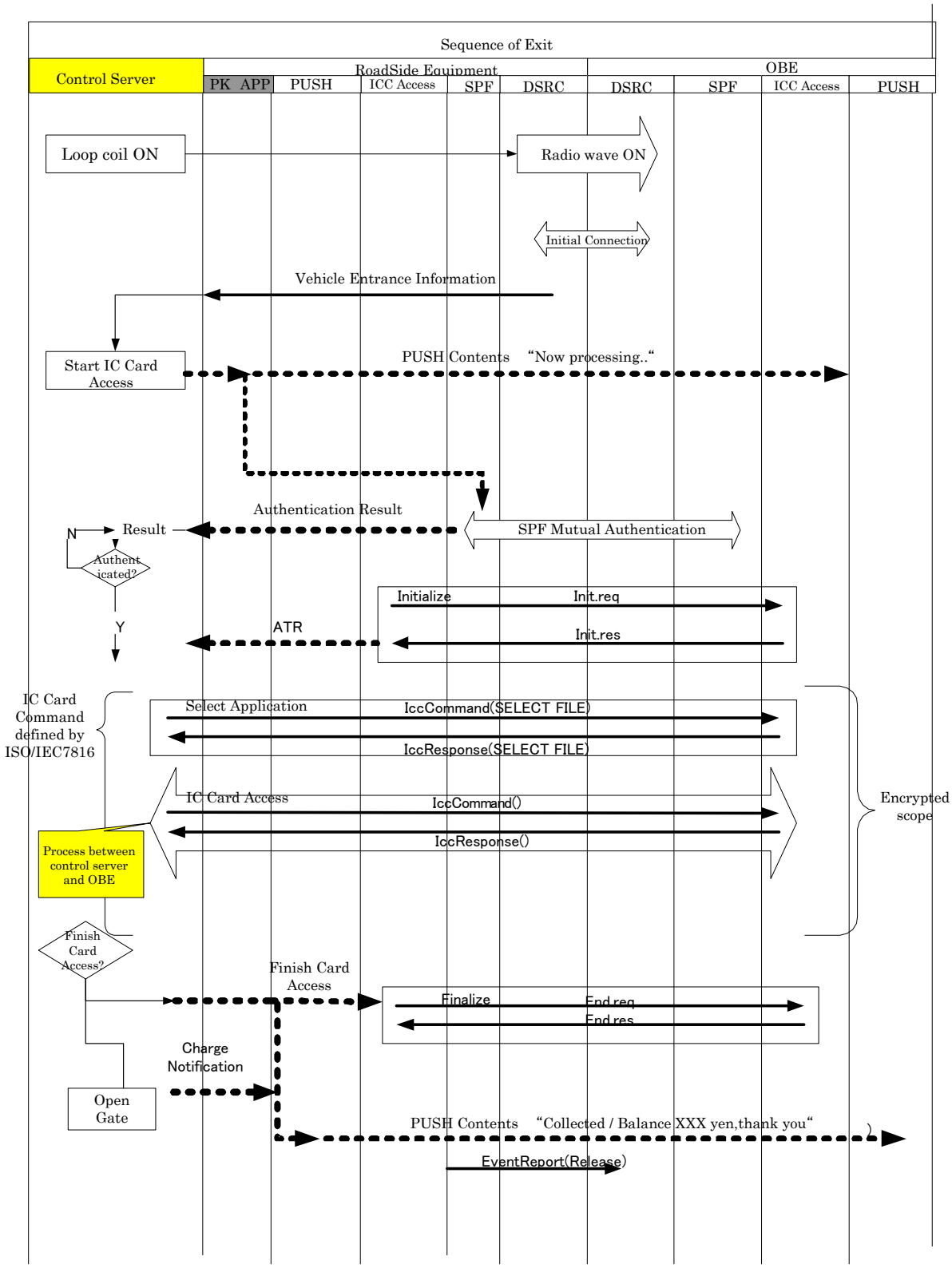


Figure C2-2 Processing Sequence of Exit

C.3 Settlement Processing using IC Card ID

The figure below shows a transaction example of the settlement processing system using the IC card access application interface.

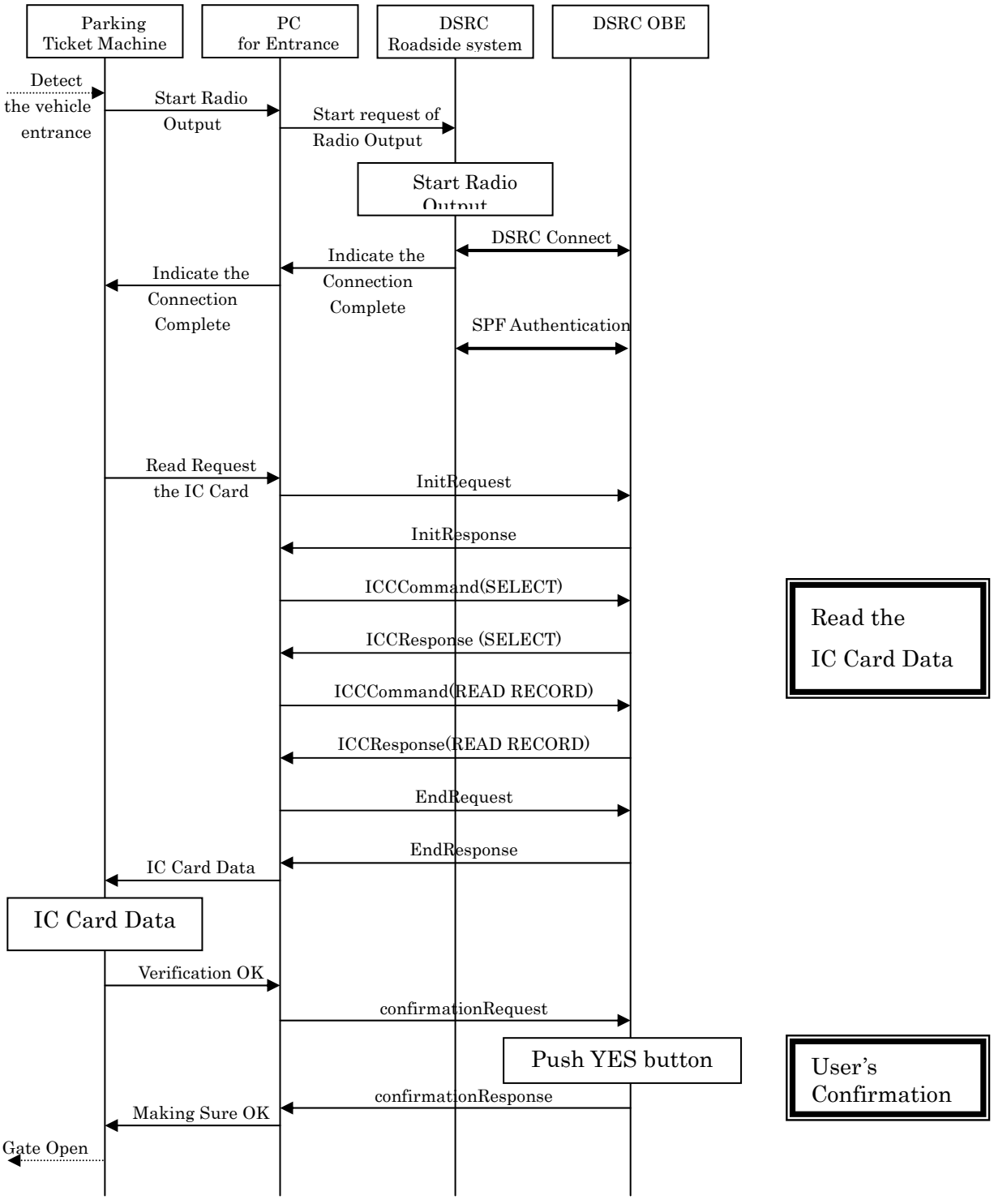


Figure C3-1 Processing Sequence of Entrance

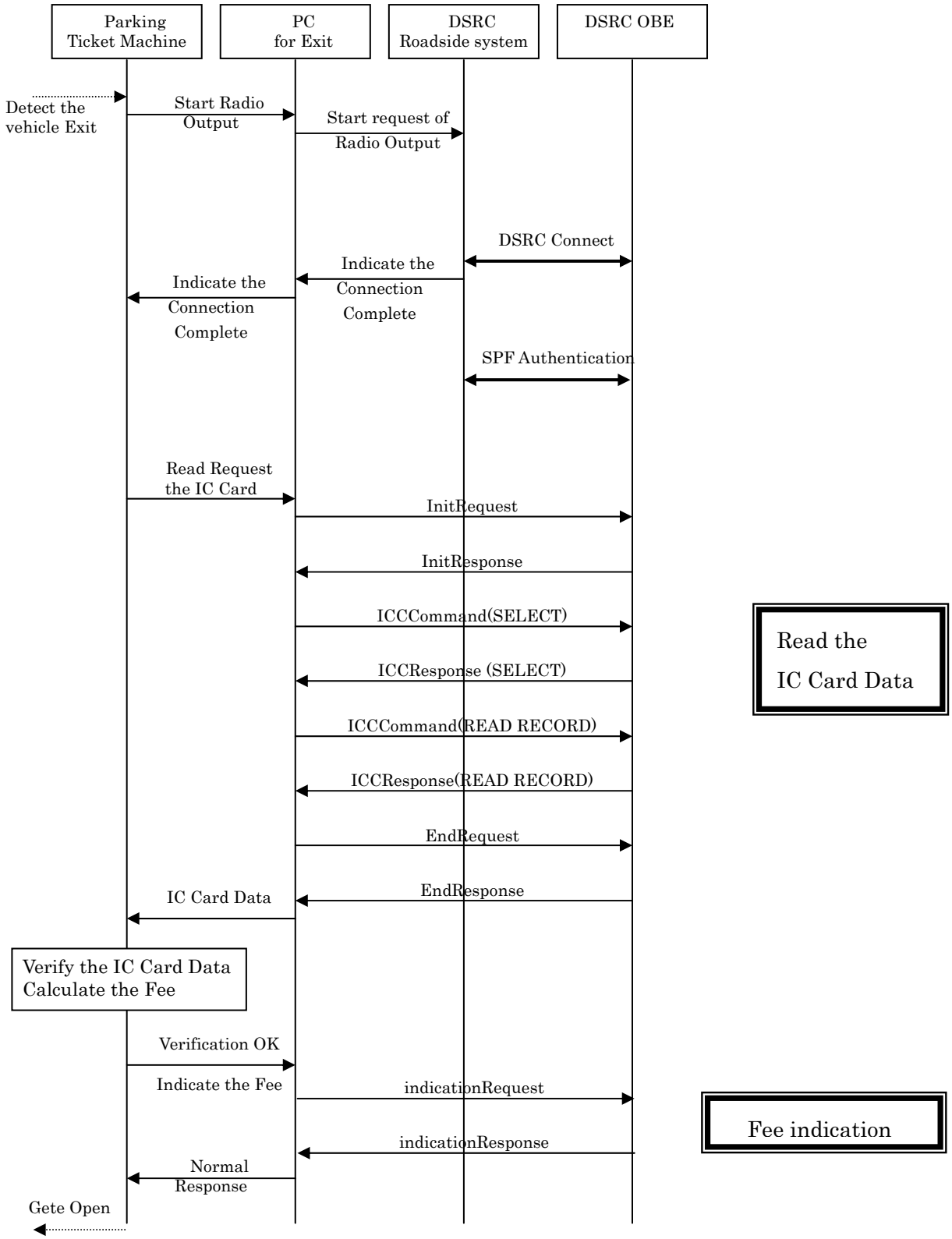


Figure C3-2 Processing Sequence of Exit

C.4 Information Providing Service using Push-type Information Delivery

(a) Point-to-point information providing service using the OBE ID

The figure below shows a transaction example in the push type information providing system using both the push type information delivery application interface and the OBE ID application interface.

In this transaction example, distributed push data is presented as voice data.

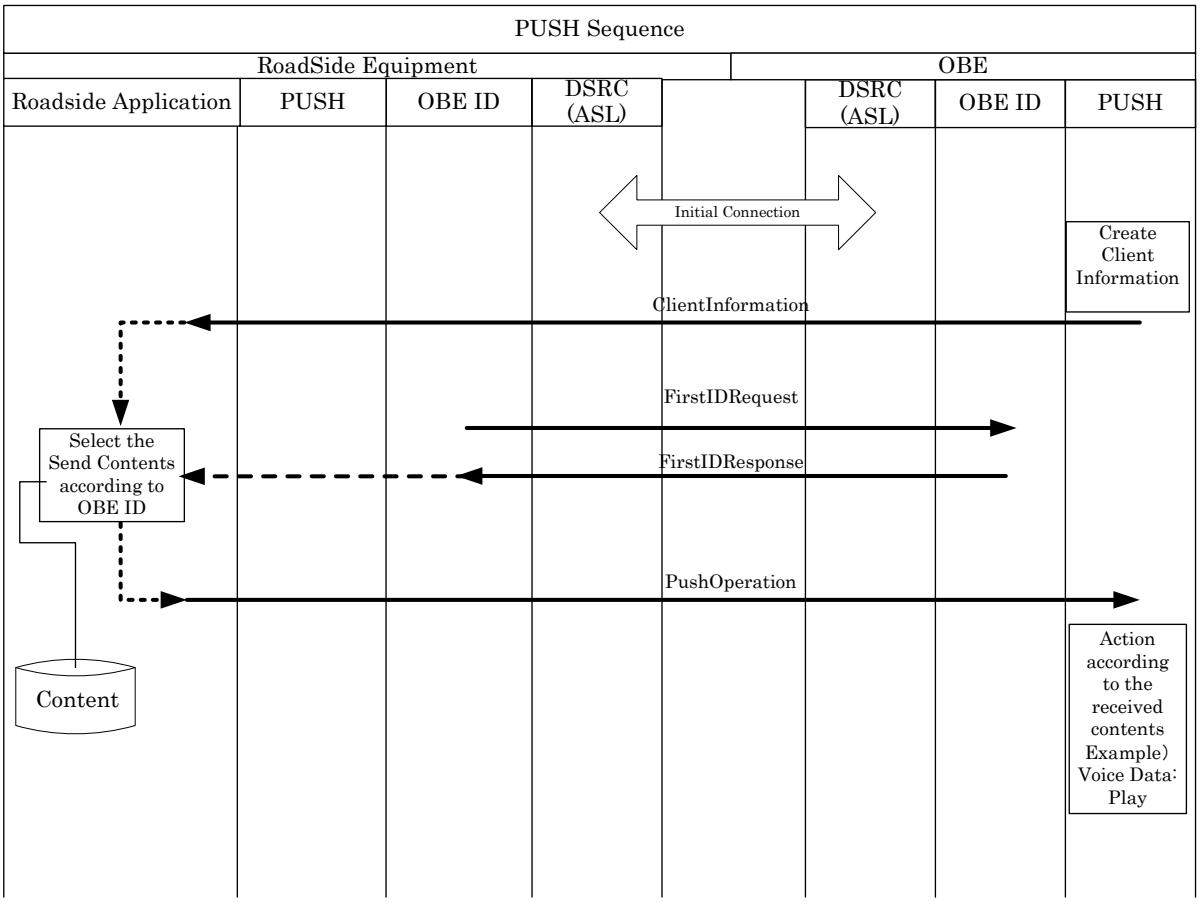


Figure C4-1 Sequence in the push type information providing service system

(b) Point-to-point information providing service for many contents

The figure below shows a transaction example in the many contents delivery service using the push type information delivery application.

In this transaction example, several voice messages are played sequentially using the push delivery function with confirmation response.

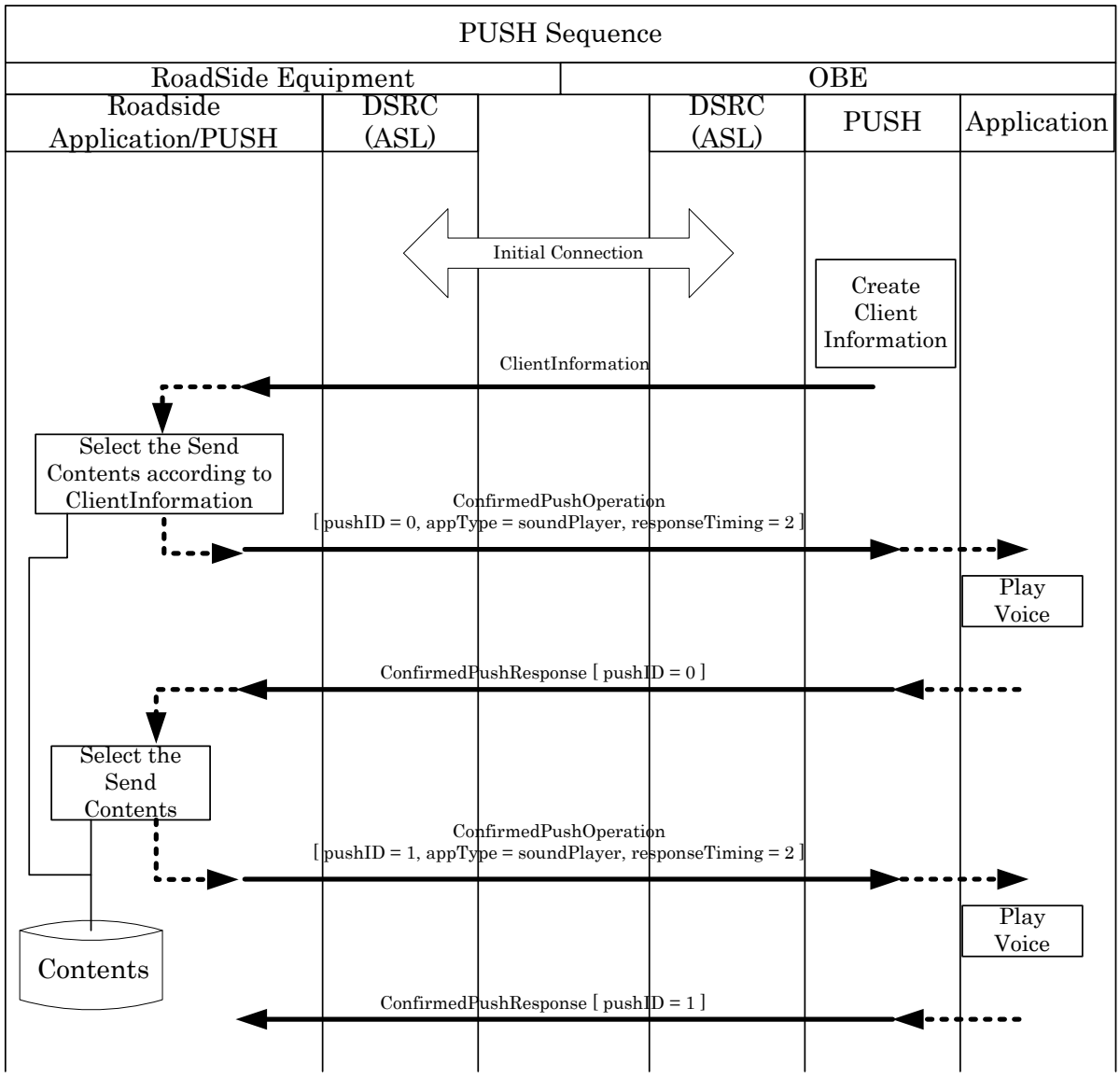


Figure C4-2 Many contents distribution sequence using the push delivery function with confirmation response

(c) Broadcast information providing service using the repeated send function

The figure below shows a transaction example in the push type information providing system using the broadcast repeated send function in the push type information delivery application.

In this transaction example, voices and images are distributed one by one.

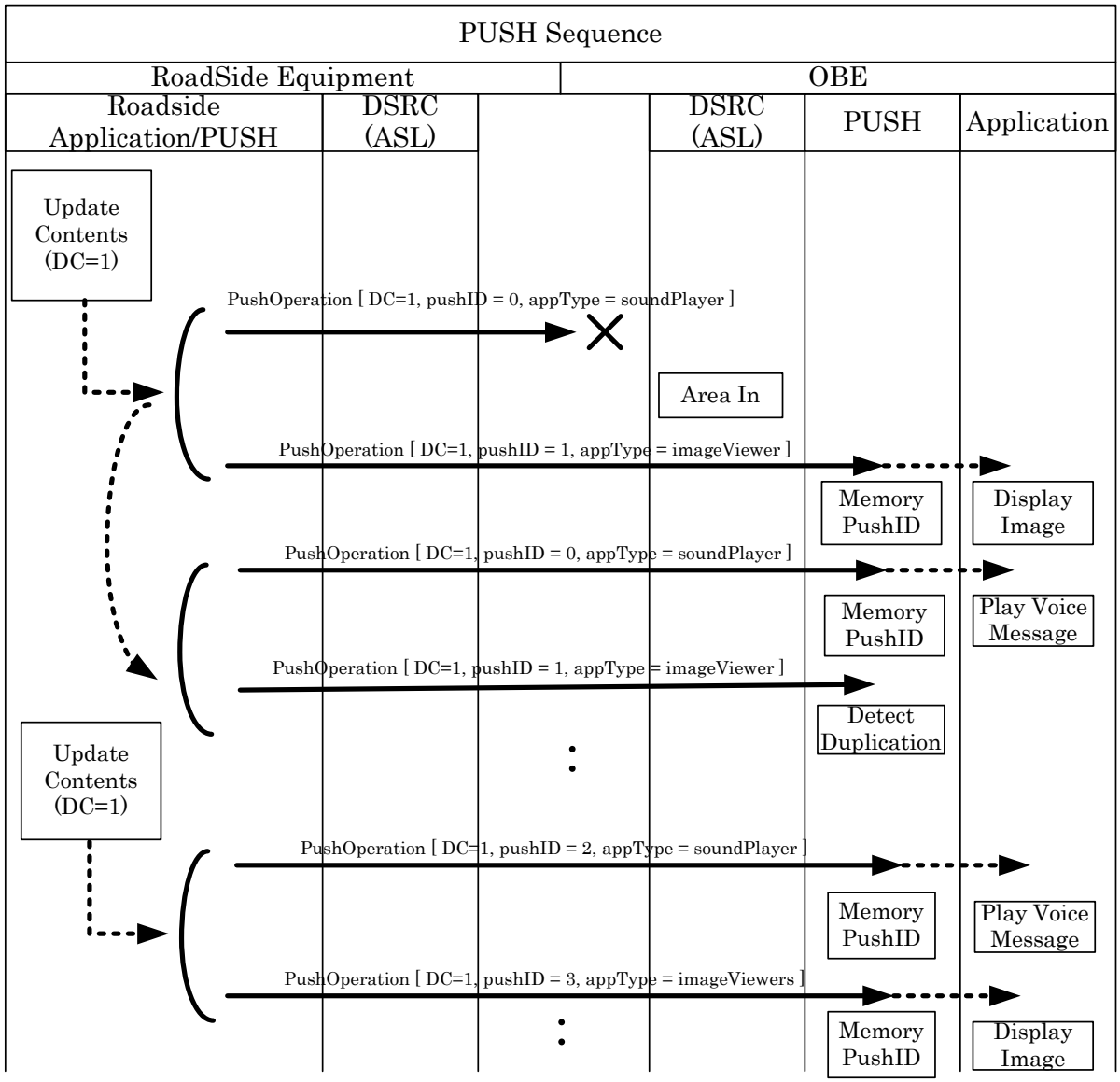


Figure C4-3 Sequence in the push type information providing service system (Broadcast, Repeat Transmission)

(d) Broadcast information providing service using the dedicated application

The figure below shows a transaction example in the push type information providing system executing the division/assembly and duplicate check in the execution application.

This transaction example shows the execution application which divides a content into segments in the roadside equipment, distributes segments, assembles segments in the OBE, and then displays images in the OBE.

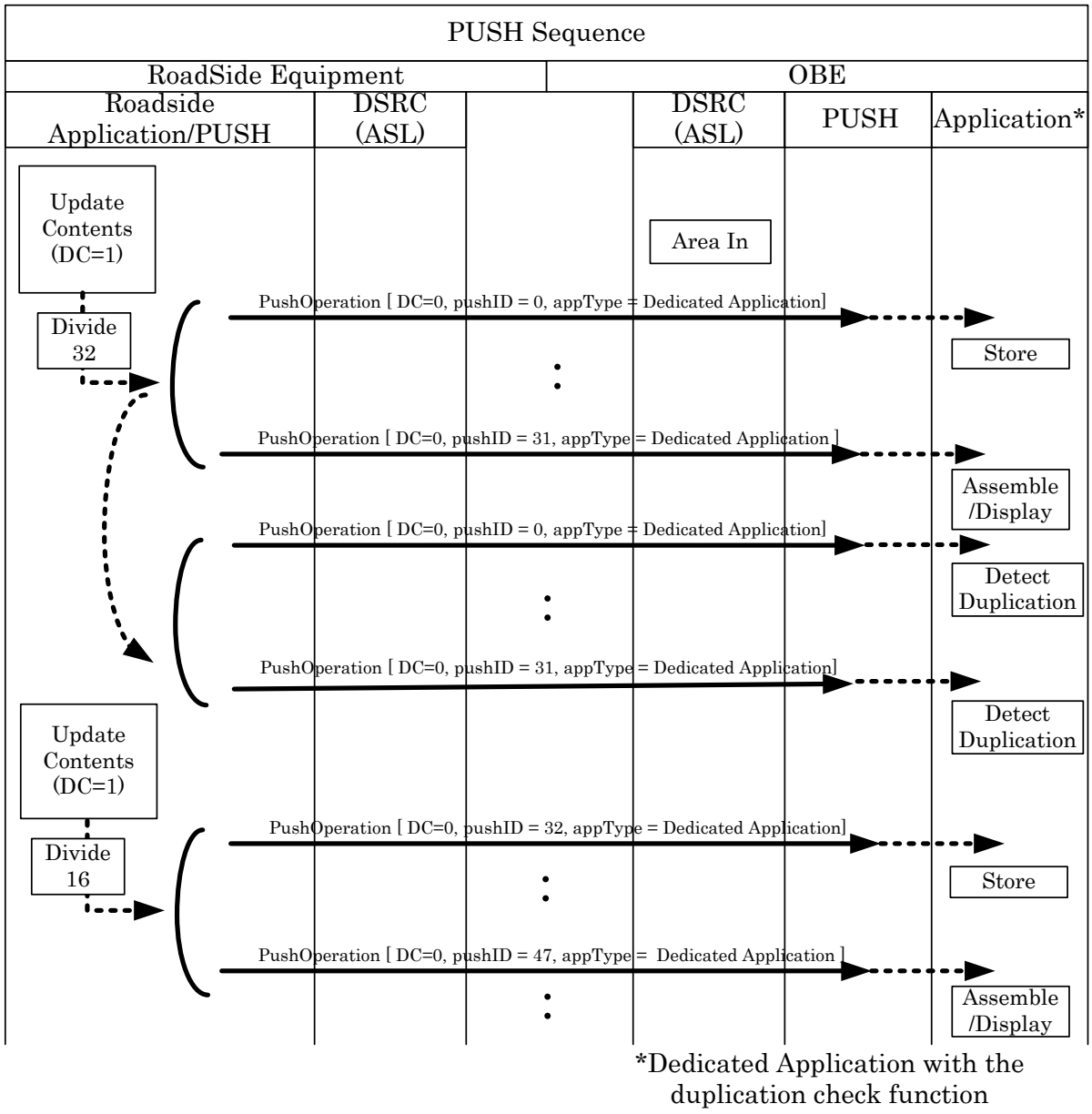
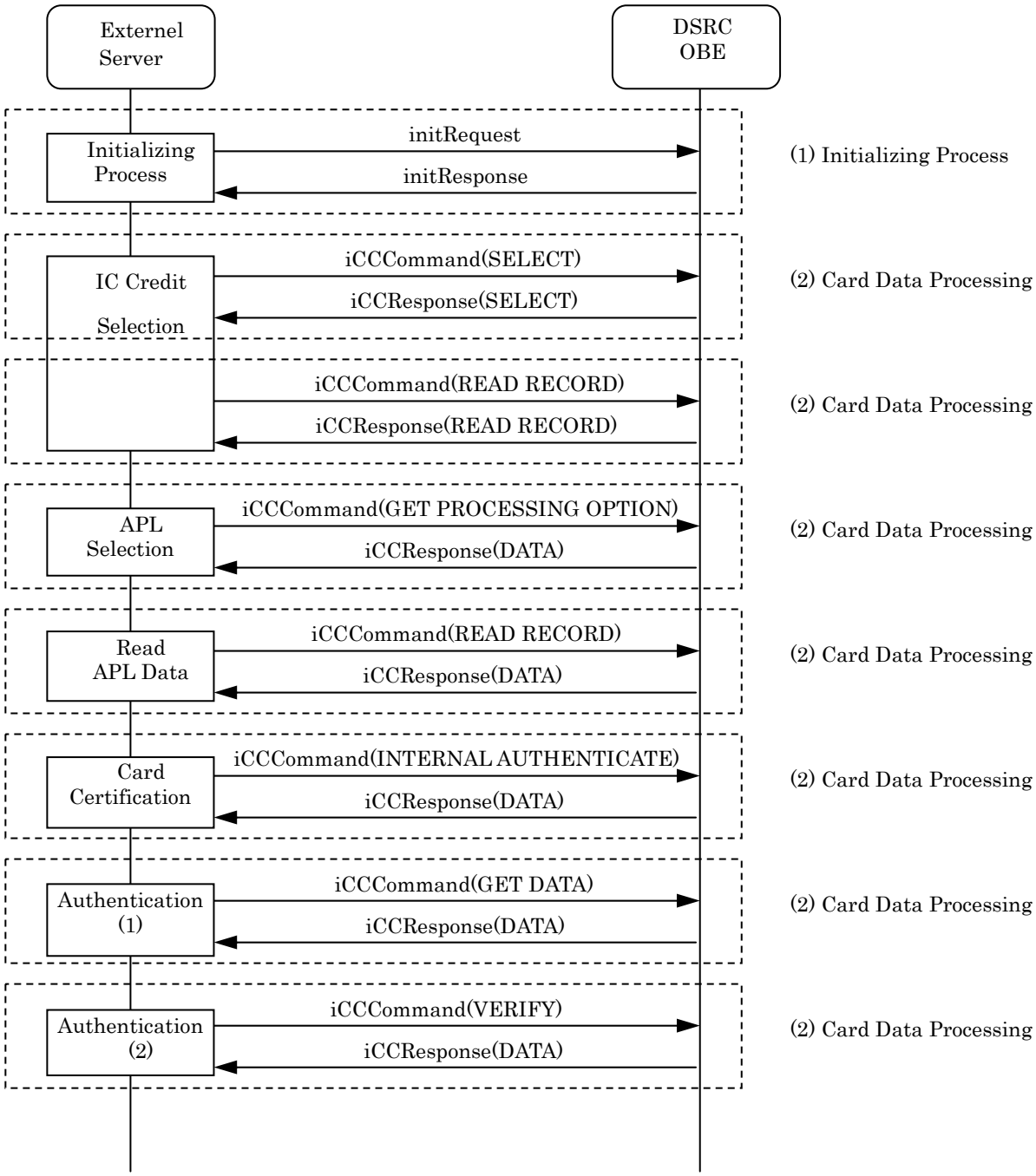


Figure C4-4 Sequence in the push type information providing service system (Broadcast, Dedicated Application)

C.5 Settlement Processing using IC Card

The figure below shows a transaction example in the settlement processing using the IC card access application interface.

Items inside parentheses in commands indicate the contents of “CommandAPDU" or “ResponseAPDU” stored in the operation data area of commands.



(Continue)

Figure C5-1 Sequence example in the settlement processing using the IC card (1/2)

(Continued)

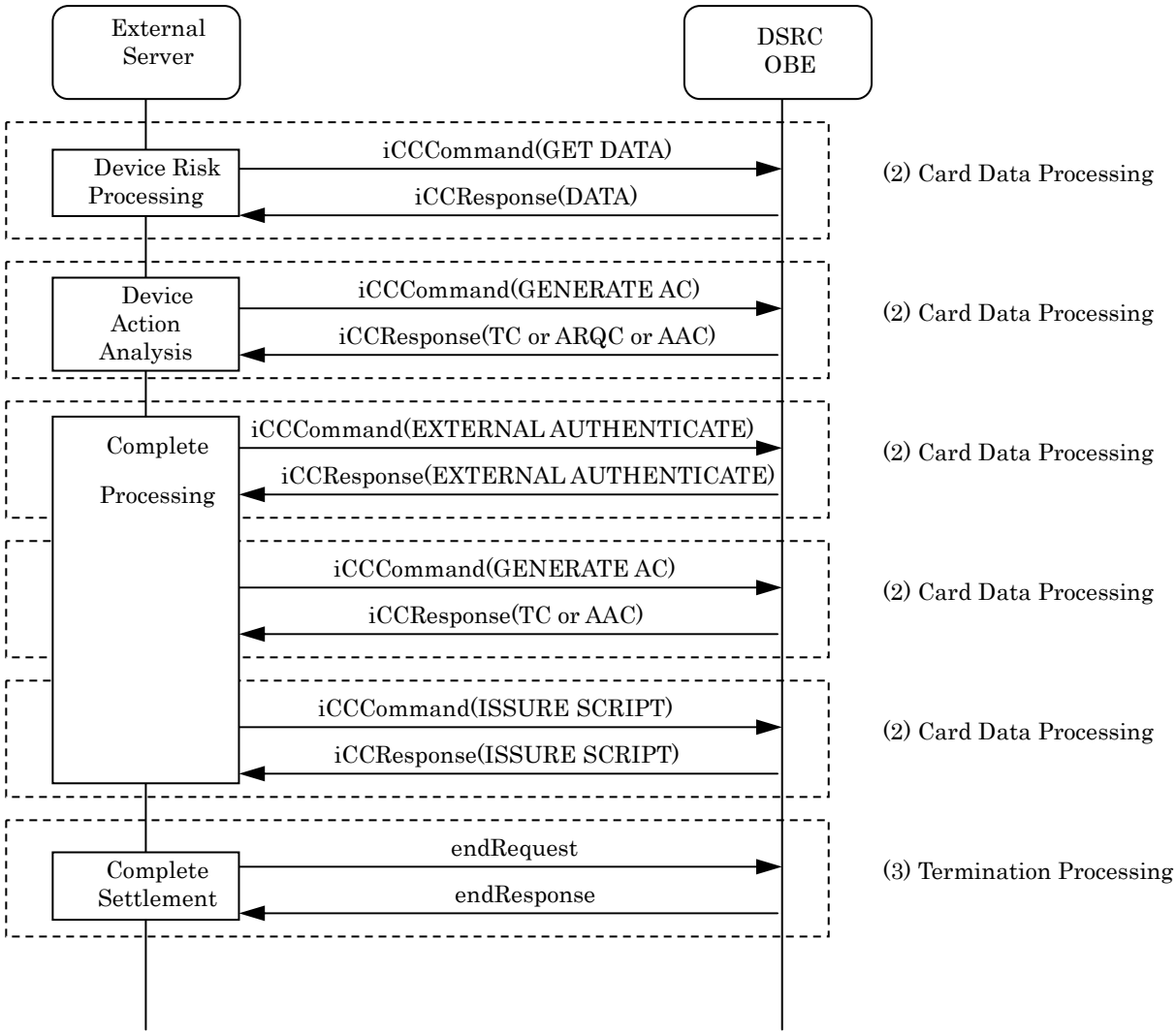


Figure C5-1 Sequence example in the settlement processing using the IC card (2/2)

Annex D OBE Memory Access Application

D.1 Memory Tag Configuration

The memory tag consists of 8 bytes.

Table D1-1 Memory Tag Format

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Memory Record Assurance	Memory Control Type	don't care (Reserved)					
2-8	Memory Tag Address (7 Octets)							

(1) Memory Record Assurance

This item indicates the storage memory attribute of data corresponding to the memory tag.

(a) Set "0" when the memory is located in the nonvolatile area.

(b) Set "1" when the memory is located in the volatile area.

(2) Memory Control Type

(a) Set "0" when the memory tag can be allocated by the roadside system.

(b) Set "1" when the memory tag is controlled by the OBE.

(3) Memory Tag Address

Set a unique value assigned to each system or provider.

The assignment method is outside the range of this guideline.

Note:Lower 6 bits of the 1st octet are handled as "don't care (reserved)". The roadside system and OBE in accordance with the current version of this guideline ignore values set in areas specified as "don't care (reserved)". In concrete, when the memory tag is "0x01", it is handled as "0x00" by the roadside system and OBE in accordance with the current version of this guideline.

D.2 Important Notice about Options

D.2.1 Memory allocation function option

The following points should be kept in mind when the OBE is not equipped with the memory allocation function and the memory free function:

(1) When receiving “memoryallocRequest” or “memoryFreeRequest” from the roadside system, the OBE shall send the OBU denial response command “obuDenialResponse” whose status is “12: No support command" to the roadside system.

(2) When receiving “readRequest” “writeRequest”, “readRequestWithCredence” or “writeRequestWithCredence” for the roadside system allocatable memory tag from the roadside system, the OBE shall send the OBU denial response command “obuDenialResponse” whose status is “6: There is no requested memory tag" to the roadside system.

(3) When receiving “readBulkRequest” “writeBulkRequest”, “readBulkRequestWithCredence” or “writeBulkRequestWithCredence” for the roadside system allocatable memory tag from the roadside system, the OBE shall regard that reading or writing of the corresponding memory tag has failed, and shall continue processing for the next memory tag.

(4) When receiving “resourceInfoRequest” from the roadside system, the OBE shall set “0” to all “storageProperty” (Property of Memory Area) of “resourceInfo” type variables.

The following point should be kept in mind when the OBE is equipped with the memory allocation function and the memory free function:

(1) When receiving “resourceInfoRequest” from the roadside system, the OBE shall set “storageProperty” (Property of Memory Area) of “resourceInfo” type variables.

D.2.2 Password option

The following points should be kept in mind when the OBE is not equipped with the password as an attribute of memory tags:

(1) Memory cannot be allocated for memory tags dedicated to the OBE having the password attribute.

(2) When receiving “memoryAllocRequestWithCredence”, “memoryFreeRequestWithCredence”, “readRequestWithCredence”, “writeRequestWithCredence”, "readBulkRequestWithCredence” or “writeBulkRequestWithCredence” from the roadside system, the OBE shall send the OBE denial response command “obuDenialResponse” whose status is “12: No support command" to the roadside system.

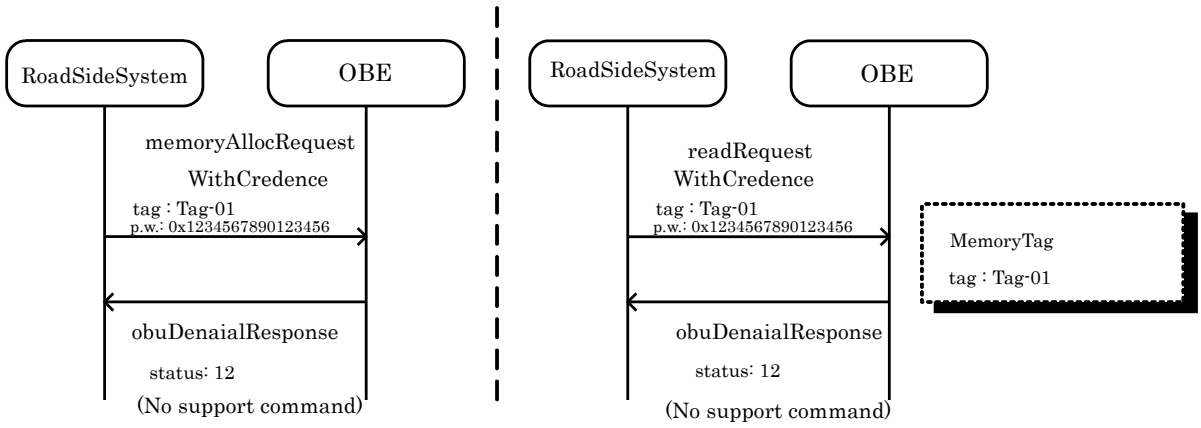


Figure D2-1 Example1 of Password Command

The following points should be kept in mind when the OBE is equipped with the password as an attribute of memory tags:

(1) When receiving “memoryAllocRequest” from the roadside system, the OBE shall allocate memory without setting the password in the memory tag. When receiving “memoryAllocRequestWithCredence” from the roadside system, the OBE shall allocate memory with setting the password in the memory tag.

(2) When receiving “memoryFreeRequestWithCredence”, “readRequestWithCredence” or “writeRequestWithCredence” for a memory tag having the password setting from the roadside system and the password disagrees, the OBE shall send the OBU denial response command “obuDenialResponse” whose status is “10: The password incompatible" to the

roadside system.

(3) When receiving “readBulkRequestWithCredence” or “writeBulkRequestWithCredence” including a memory tag having the password setting from the roadside system and the password disagrees, the OBE shall regard that reading or writing of the corresponding memory tag has failed, and shall continue processing for the next memory tag.

(4) When receiving “memoryFreeRequest”, “readRequest” or “writeRequest” for a memory tag having the password setting from the roadside system, the OBE shall send the OBU denial response command “obuDenialResponse” whose status is “10: The password incompatible" to the roadside system.

(5) When receiving “readBulkRequest” or “writeBulkRequest” including a memory tag having the password setting from the roadside system, the OBE shall regard that reading or writing of the corresponding memory tag has failed, and shall continue processing for the next memory tag.

(6) When receiving “memoryFreeRequestWithCredence”, “readRequestWithCredence” or “writeRequestWithCredence” for a memory tag not having the password setting from the roadside system, the OBE shall send the OBU denial response command “obuDenialResponse” whose status is “10: The password incompatible" to the roadside system.

(7) When receiving “readBulkRequestWithCredence” or “writeBulkRequestWithCredence” including a memory tag not having the password setting from the roadside system, the OBE shall regard that reading or writing of the corresponding memory tag has failed, and shall continue processing for the next memory tag.

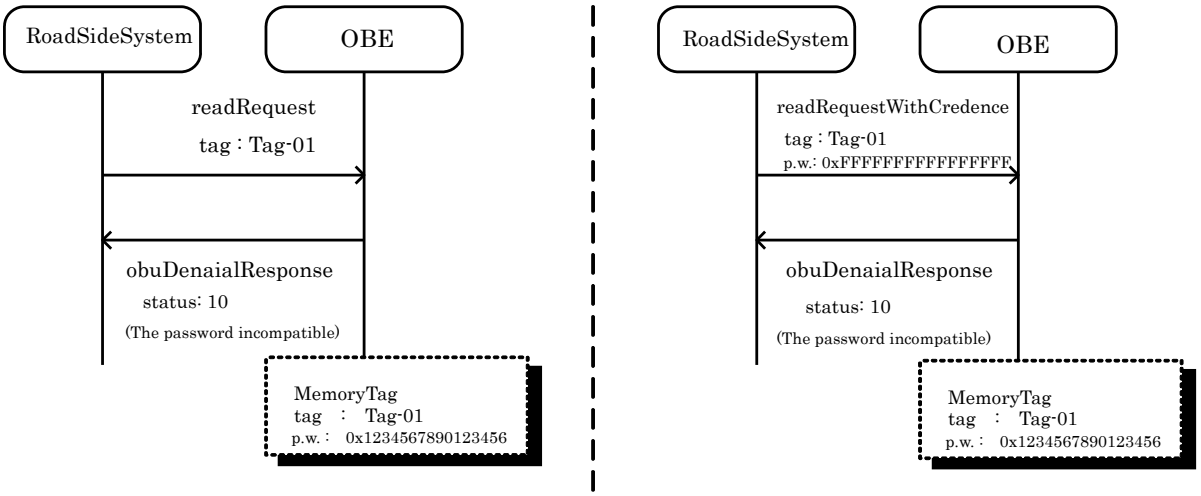


Figure D2-2 Example2 of Password Command

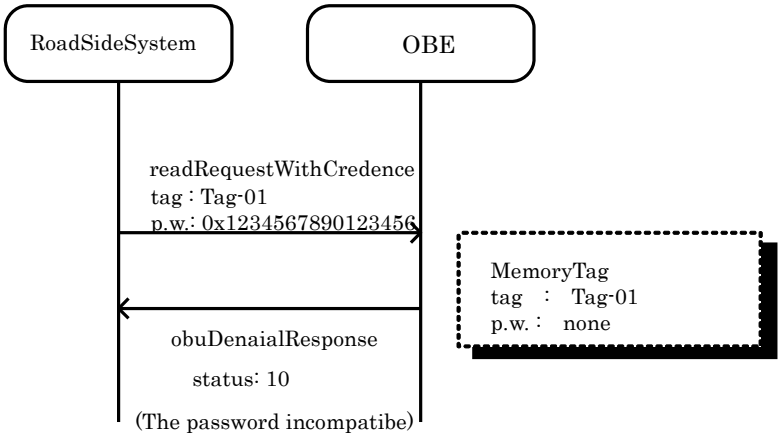


Figure D2-3 Example3 of Password Command

D.3 Protection Mode

The memory access application has two ports, a port using the DSRC-SPF (secure port LP3) and a port not using the DSRC-SPF (normal port LP2) as described in Annex. B. When allocating a memory area, select whether or not the DSRC-SPF is essential.

For memory tags for which the SPF is essential, the OBE accepts only commands by way of the LP3, and sends the OBE denial response command "obuDenialResponse" whose status is "11: SPF violation" to the roadside system for commands by way of the LP2. For memory tags for which the SPF is inessential, the OBE accepts commands by way of the LP2 and commands by way of the LP3.

Note that memory allocation with the setting "SPF in Protection Mode essential" is disabled for the OBE not having the LP3.

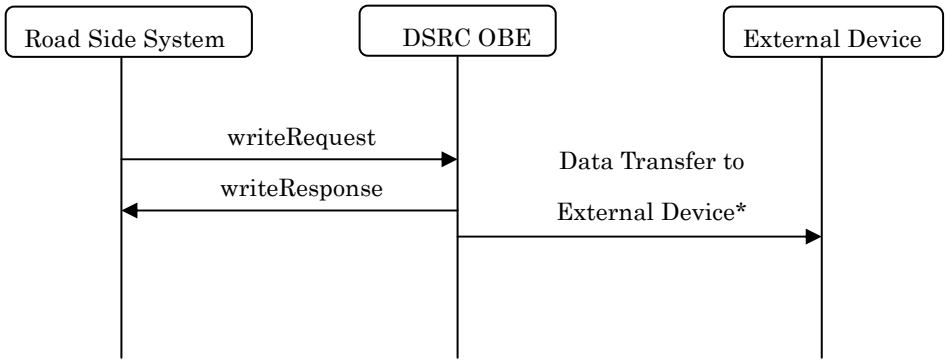
D.4 Memory Access Management

D.4.1 Management of OBE controlling memory

D.4.1.1 Application to data exchange with external device

Memory tags are determined in advance between the roadside system and the OBE for the OBE controlling memory. The OBE controlling memory is available for reading and writing data stored in the OBE, and exchanging data between the OBE and the external device (such as vehicle navigation equipment and mobile telephone) connected to the OBE.

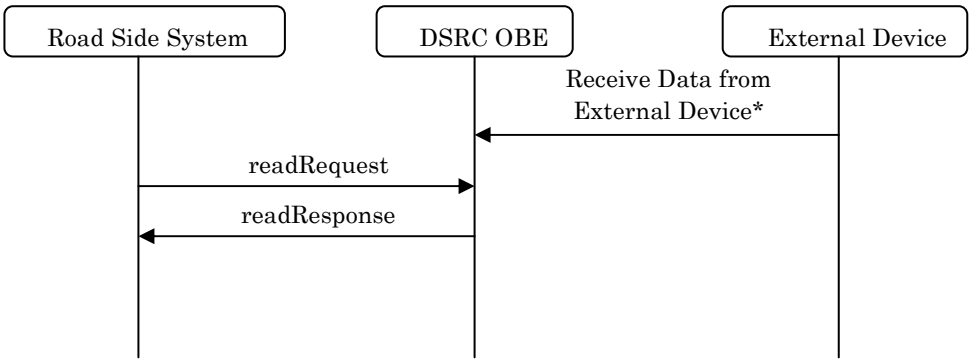
This guideline does not specify the data transfer method and timing with the external device. When executing data exchange with the external device, however, required items shall be determined in advance between the roadside system and the OBE. Examples are shown below.



Note:

The data sending method and timing to the external device are not specified because they depend on the external device.

Figure D4-1 Data sending to the external device using the OBE controlling memory



Note:The data receiving method and timing from the external device are not specified because they depend on the external device.When data has not reached (or is being received) from the external device when the OBE receives “readRequest”, the OBE immediately gives the response “no data”, or may delay giving “readResponse” until data receiving is finished. The actual response to be given depends on the specification determined for each data between the roadside system and the OBE.

Figure D4-2 Data receiving from the external device using the OBE controlling memory

D.4.1.2 Application example of OBE controlling memory to uplink

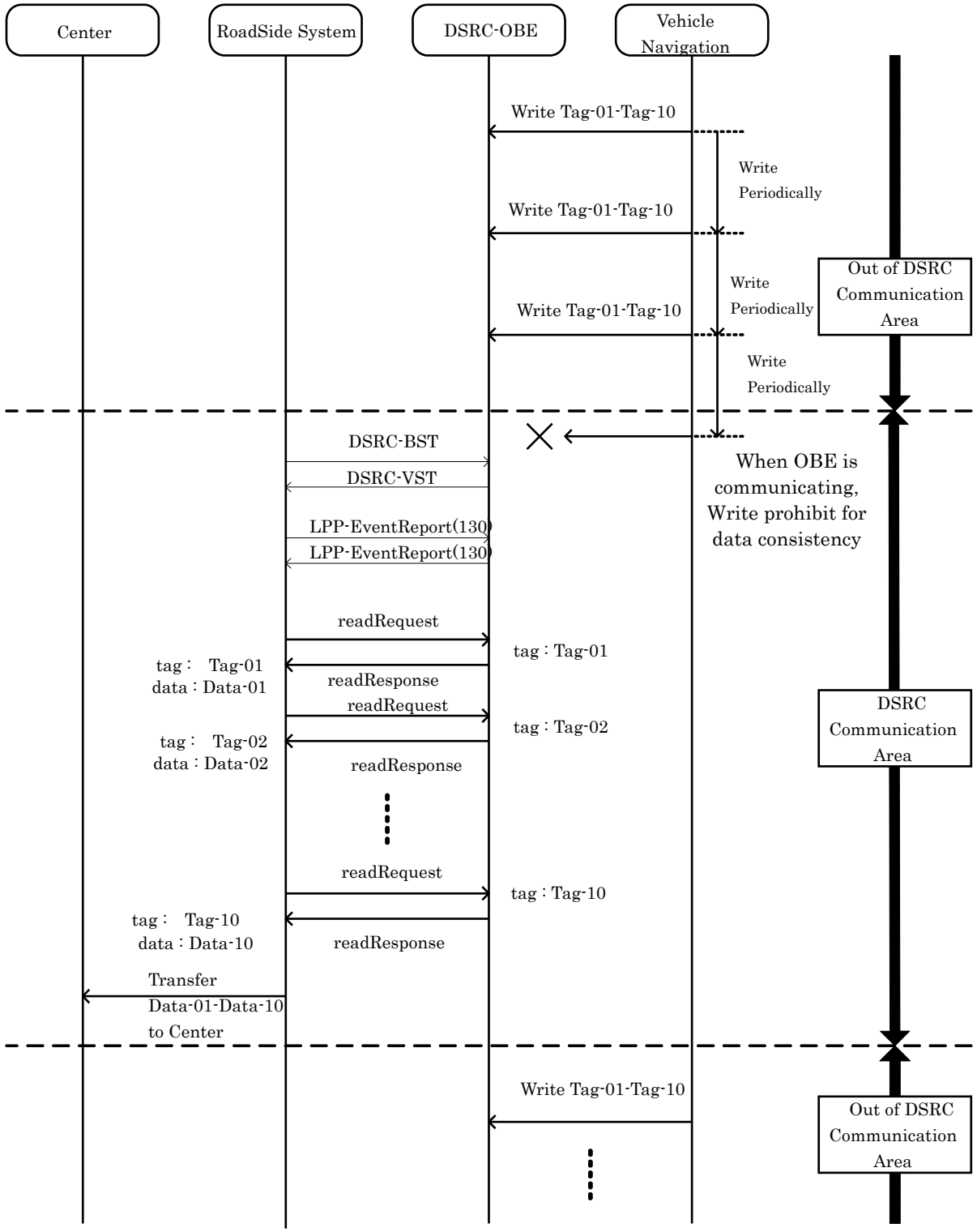
The figure below shows an example in which the travel history information from the vehicle navigation equipment is uploaded using the OBE controlling memory.

Table D4-1 shows an example of allocating the memory, and Figure. D4-3 shows a sequence example.

This guideline does not specify the data transfer method and timing with the vehicle navigation equipment. When several tags make sense as some information, however, attention should be paid to ensure consistency so that tag information being uploaded is not overwritten by other tag information.

Table D4-1 Example of OBE controlling memory tag

Tag	Protection Mode	Password	Memory allocation size	Content
Tag-01	ReadOnly/without SPF	without	250 bytes	current driving position
Tag-02	ReadOnly/without SPF	without	250 bytes	driving history data1
:	:	:	:	:
Tag-10	ReadOnly/without SPF	without	250 bytes	driving history data9



Note:Data-** is data corresponding to Tag-**

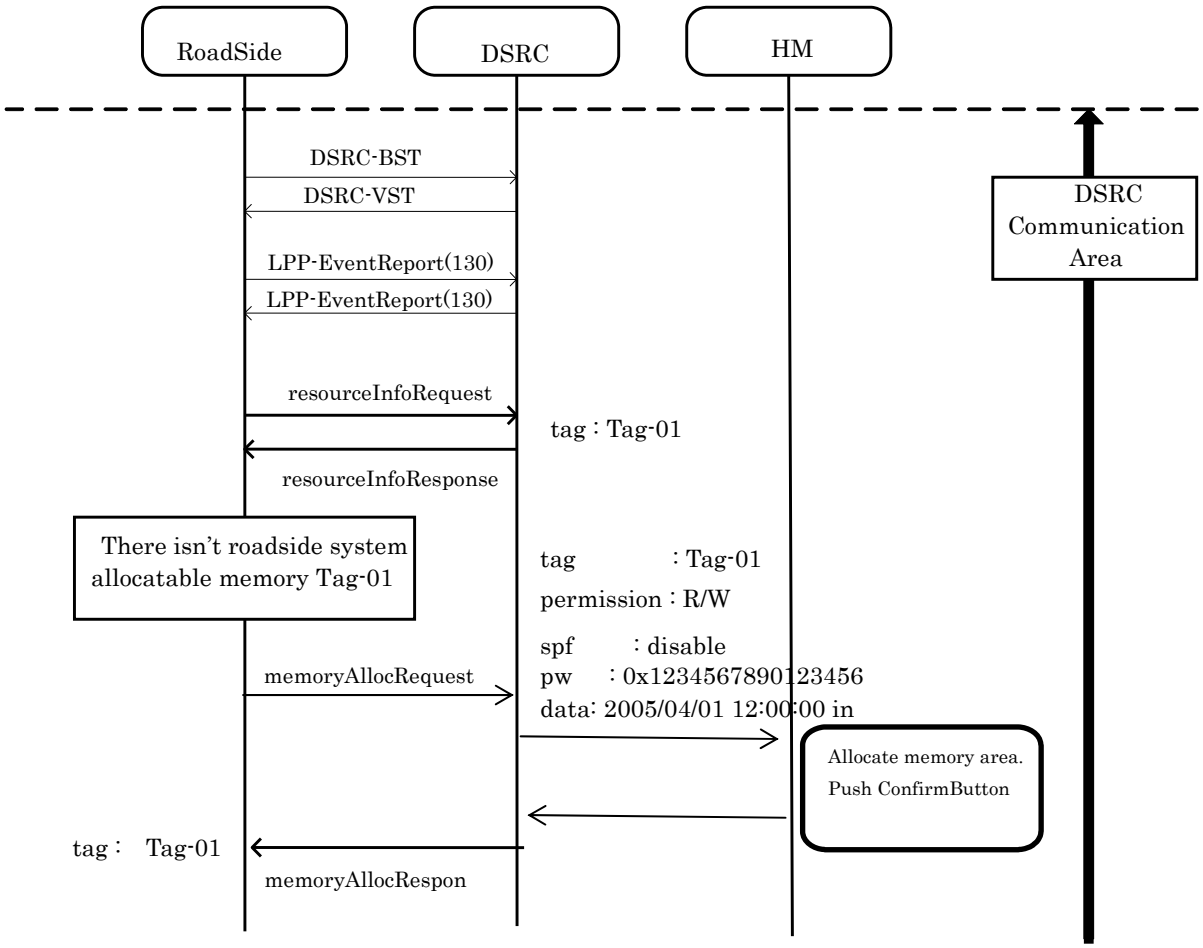
Figure D4-3 Example of upload sequence in the OBE using the OBE controlling memory

D.4.2 Management of roadside system allocatable memory

This guideline postulates that use of the roadside system allocatable memory is started when the roadside system allocatable memory area dedicated to the provider is allocated in the DSRC OBE based on a contract or agreement between the provider and the user (OBE purchaser). After that, the provider manages the system in conformity to laws and regulations such as the Private Information Protection Law.

It is preferable that the OBE handling memory allocation request commands has the function to confirm the user's intention when receiving such commands. The user's intention confirmation function depends on the manufacturer's arbitrary specification, and is outside the range of this guideline.

Figure. D4-4 and D4-5 show examples of memory writing sequence using the roadside system allocatable memory. Figure. D4-4 shows a case in which the OBE has the function to confirm the user's intension when allocating the memory.



FigureD4-4 Example of memory writing sequence using the roadside system allocatable

memory (1st time)

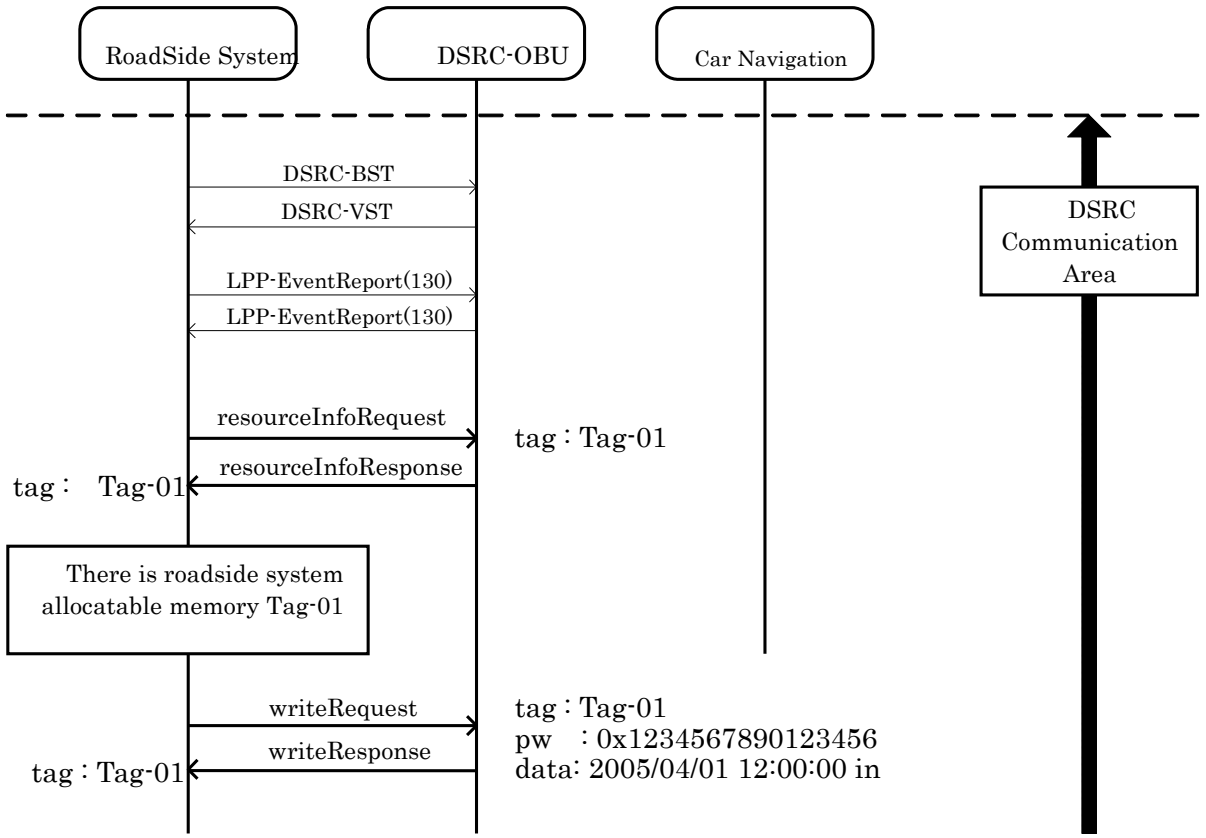


Figure D4-5 Example of memory writing sequence using the roadside system allocatable memory (after2nd time)

Data written to the OBE memory should be information encrypted by the provider. It is preferable that encrypted data cannot be decrypted by the user or third parties. It is preferable also that the OBE can delete memory tags and stored data without using the password for keeping out malicious providers and using resources effectively.

This management example for the roadside system allocatable memory postulates a consolidated framework which can be confirmed by the OBE user (through access to website, etc.) with regard to the uniqueness of roadside system allocatable memory tags and linking between the memory allocated provider and the roadside system allocatable memory tags (linking between memory tags and the provider name who allocated the memory).

Annex E Implementation Example and Important Notice about OBE ID Communication Application

This data shows an example of adopting the (library used in the) DSRC-SPF as the in-application security in the OBE ID communication application, and describes important notice when the in-application security is handled and when the in-application security is not handled. The setup method is outside the range of this guideline, and shall be specified separately.

E.1 Example when DSRC-SPF is adopted as In-application Security

This example adopts the DSRC-SPF as the in-application security in the OBE ID communication application.

E.1.1 Positions and functions of in-application security and DSRC-SPF

Figure. E1-1 shows the positions of the DSRC-SPF and in-application security in the OBE ID communication application. The DSRC-SPF intends authentication and cryptographic communication among equipment, and cannot offer access control different for each provider.

On the other hand, the execution method and key issue can be selected and determined for each provider for the in-application security. Accordingly, it is possible to execute different access control and cryptographic communication for each provider. The DSRC-SPF (library) is available as the in-application security. At this time, the used SPF and parameters for the SPF are selected using the security profile set up in the OBE.

The security profile indicates data on the SPF type and parameters used for the SPF. The security profile is linked with the acquirer ID, and the used SPF and parameters used for the SPF are selected in accordance with the security profile. The security profile is specified respectively for each SPF.

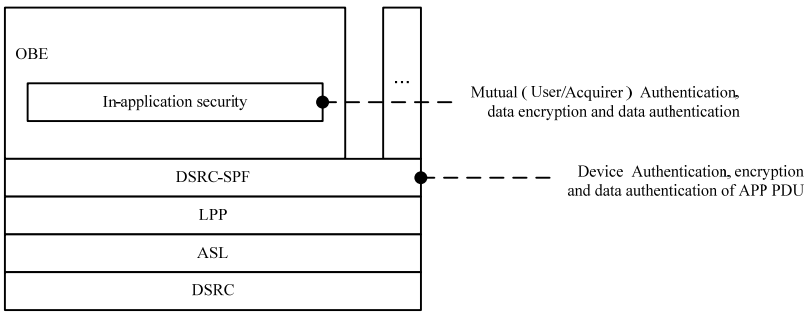


Figure E1-1 Positions of in-application security and DSRC-SPF

E.1.2 Concrete specification

This paragraph describes concrete specification when the DSRC-SPF is adopted as the in-application security in the OBE ID communication application.

E.1.2.1 Command format

This paragraph describes the detailed definition of “AuthenticateCommand” type variables (authPath1 to authPath4) and use method of “SecondIDRequest” type variables.

E.1.2.1.1 authPath1(NegotiateRequest)

NegotiateRequest of DSRC-SPF is set as authPath1.

Table E1-1 authPath1 Command Format

	7(MSB)	6	5	4	3	2	1	0
1	version				fill(0)			
2	Command Type authenticateCommand(0)							
3	Operation Type authPath1(0)							
4	Length of NegotiateRequest							
	NegotiateRequest							

E.1.2.1.2 authPath2(NegotiateResponse)

NegotiateResponse of DSRC-SPF is set as authPath2

Table E1-2 authPath2 Command Format

	7(MSB)	6	5	4	3	2	1	0
1	version				fill(0)			
2	Command Type authenticateCommand(0)							
3	Operation Type authPath2(1)							
4	Length of NegotiateResponse							
	NegotiateResponse							

E.1.2.1.3 authPath3 (SetupMessageRequest)

SetupMessageRequest of DSRC-SPF is set as authPath3.

Table E1-3 authPath3 Command Format

	7(MSB)	6	5	4	3	2	1	0
1	version				fill(0)			
2	Command Type authenticateCommand(0)							
3	Operation Type authPath3(2)							
4	Length of SetupMessageRequest							
	SetupMessageRequest							

E.1.2.1.4 authPath4 (SetupMessageResponse)

SetupMessageResponse of DSRC-SPF is set as authPath4.

Table E1-4 authPath4 Command Format

	7(MSB)	6	5	4	3	2	1	0
1	version				fill(0)			
2	Command Type authenticateCommand(0)							
3	Operation Type authPath4(3)							
4	Length of SetupMessageResponse							
	SetupMessageResponse							

E.1.2.1.5 “SecondIDResponse” type variable use method

```
SecondIDResponse::=SEQUENCE{
    encryptionAlgorithmId        INTEGER(0..255),
        -- Cryptographic algorithm to keep the ID confidential
        -- (for in-application security)
    keyNumber                    INTEGER(0..255),
        --Key number to keep the ID confidential (for in-application security)
    encryptedId                  OCTET STRING
        -- Encrypted ID information (for in-application security)
}
```

Values of the variables “encryptionAlgorithmId” and “keyNumber” are specified separately for each SPF. The variable "encryptedId” stores “SpfPDU" of the DSRC-SPF.

E.1.2.1.6 Denial response status

Table E1-6 shows the status notified in the denial response used for the in-application security. The status code shown in the table below subtracted by “32” is equivalent to the status code of the DSRC-SPF. However, "32” and “255” are used for “Authentication not completed” and “Other error inside OBE” respectively.

Table E1-6 Contents of “status” in the negative acknowledgment for the in-application security

Status code	Remarks
33 (32+1)	Security type error
34-35	For future use
36 (32+4)	DSRC-SPF version disagreed
37 (32+5)	SPF internal error
38-47	For future use
48 (32+16)	Illegal Service Primitive
49	For future use
50 (32+18)	Provider identifier not supported
51-63	For future use

E.1.2.2 Sequence example

Figure. E1-2 shows a sequence example when the DSRC-SPF is adopted as the in-application security.

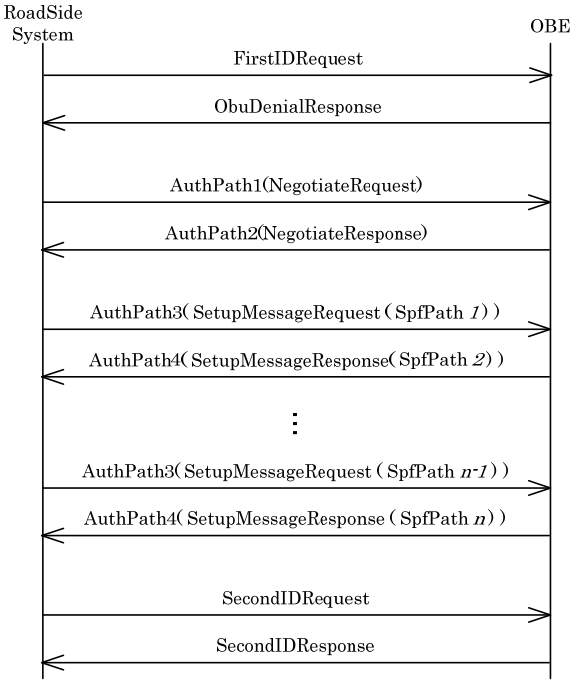


Figure E1-2 Sequence Example

E.1.2.3 ID acquisition procedure when DSRC-SPF is adopted as in-application security

This paragraph describes the procedure when the DSRC-SPF is adopted as the in-application security.

(1) The roadside system sends “firstIDRequest” to the OBE.

(2) When receiving “firstIDRequest”, the OBE refers to the ID registration information, and sends “firstIDResponse” or “obuDenialResponse” to the roadside system as follows:

(a) When the condition of the OBE ID corresponding to the acquirer ID is “plaintextIDRefusal (false (0))”, the OBE sends “firstIDRequest” to the roadside system. When the condition is “plaintextIDRefusal (true (1))”, the OBE sends “obuDenialResponse” whose status is "32: Plain text send refused, authentication failed or no authentication” to the roadside system.

(b) When the OBE ID is not registered at all in the OBE, the OBE sends “obuDenialResponse” whose status is “12: OBE ID not registered at all” to the roadside system.

(c) When the OBE ID corresponding to the specified acquirer is not registered, the OBE sends “obuDenialResponse” whose status is “2: No OBE ID corresponding to acquirer ID” to the roadside system.

(3) When receiving “obuDenialResponse” whose status is “32” from the OBE in the processing in the step (2)-(a), the roadside system sends “authPath1(NegotiateRequest)” to the OBE for notifying the list of parameters available in the DSRC-SPF.

(4) When receiving “authPath1(NegotiateRequest)”, the OBE refers to the security profile corresponding to the acquirer ID specified in “firstIDRequest”, and sends “authPath2(NegotiateResponse)” or “obuDenialResponse” to the roadside system as follows:

(a) The OBE selects a parameter to be used from the parameter list specified in “authPath1(NegotiateRequest)”, and sends “authPath2(NegotiateResponse)” to the roadside system for notifying the selected parameter.

(b) When no available parameter is included in the parameter list specified in “authPath1(NegotiateRequest)”, the OBE sends “obuDenialResponse” whose status is “33: Security type error” to the roadside system.

(5) When receiving “authPath2(NegotiateResponse)” from the OBE in the processing in the step (4)-(a), the roadside system acquires the parameter specified by the OBE in “authPath2(NegotiateResponse)”.

(6) The roadside system creates a request using the parameter specified by the OBE, and sends “authPath3(SetupMessageRequest)” to the OBE.

(7) When receiving “authPath3(SetupMessageRequest)”, the OBE creates a response using the specified parameter, and sends “authPath4(SetupMessageResponse)” to the roadside system.

(8) The processing in the steps (6) and (7) are repeated until the SPF sequence is finished.

(9) When the SPF sequence is finished, the roadside system sends “secondIDRequest” to the OBE.

(10) When receiving “secondIDRequest”, the OBE refers to the ID registration information, and sends “secondIDResponse” or “obuDenialResponse" to the roadside system as follows:

(a) When the OBE ID condition corresponding to the acquirer ID agrees with the result acquired by the processing in the steps (4) to (8), the OBE sends “secondIDResponse” to the roadside system.

(b) When the OBE ID condition corresponding to the acquirer ID does not agree with the result acquired by the processing in the steps (4) to (8), the OBE sends “obuDenialResponse” whose status is "32: Plain text send refused, authentication failed or no authentication” to the roadside system.

E.2 Important Notice when In-application Security is not handled

When the OBE not handling the in-application security receives the following command, it sends back "ObuDenialResponse" whose status is "32".

(1) AuthenticateCommand

(2) secondIDRequest

To the ID condition (IDCondition), set values indicating that the in-application security is not used.

E.3 Important Notice for handling In-application Security

The in-application security shall be selected or determined properly on the responsibility of the provider.

Keep the following points in mind when handling the in-application security.

(1) ID condition

In the "ciphertextIDRefusal" and "mutualAuthentication" fields indicating the ID condition (shown in Table 3.5-1), set the necessity of encryption and authentication in advance in accordance with the provider's policy. When setting the necessity through DSRC communication, use the ID condition change request command (maintenance command). Use of other means is not hindered.

(2) Data authentication

When data authentication is required for the OBE ID, use "mACForOriginalText" in "ObuID" (described in "3.5.3 Definition of data configuration".)

(3) AuthenticateCommand

Use "AuthenticateCommand" ("IDAcquisitionCommand" described in "3.5.3 Definition of data configuration") to exchange required information and perform mutual authentication between the roadside system and the OBE based on the specification of the mounted in-application security.

(4) SecondIDResponse

Determine the contents of "SecondIDResponse" through exchange of information required in the security executed by "AuthenticateCommand".

(5) OBE denial response

The definition of status 32 to status 63 used in "obuDenialResponse" (shown in Table 3.5-9) is provided for the in-application security. Use the status 33 to 63 in accordance

with the specification of in-application security.

E.3.1 When there are several providers and several in-application securities

The following two cases are postulated:

(a) When one provider uses one or more in-application securities

(b) When one or more providers share a same in-application security

In the case (a), it is preferable that the OBE provides proper firewall among securities, has the function to control the correspondence between the acquirer ID and several in-application securities, and specifies the in-application security selection procedure using "AuthenticateCommand". The concrete specification shall be specified for each in-application security.

In the case (b), it is preferable that the OBE has the function to control the correspondence between the in-application security and several acquirer IDs.

Annex F Version of Basic Application Interface

F.1 Definition

The version of the basic application interface (API) means a unique combination of available API for additional definitions of the basic API and specification update of the basic API.

F.2 Intended Purpose

The version of the basic API is used to select the basic API by the roadside system in a session between the roadside system and the OBE.

Note:Roadside system should have the mechanism to select applicable the basic API in accordance with the architecture of the basic API

F.3 Elements of Version Management

F.3.1 Version management table

The version management table is used to uniquely determine a combination of the basic API used in each version. This management table is stored local port numbers assigned to the basic API used in the corresponding version.

The roadside system shall have the version management able corresponding to the installed version of the basic API.

F.3.2 Version number

The version number is the identifier stored in the basic API commands. The version number shall be "1".

F.4 Version Update

The version of the basic API is updated in units of API. When the basic API version is updated, local port numbers assigned to the basic API shall be also updated.

Table F.4-1 shows version update and the contents of version management tables.

Version 2 show update of the API1. Version 3 show update of the API2 and API3. Version 4 addition show of the API5.

Table F.4-1 Version update and contents of version management tables

Version of Basic API	Version Management Table				
	API1	API2	API3	API4	API5
4(New)	0x0001	0x0011	0x0021	0x0030	0x0040
3	0x0001	0x0011	0x0021	0x0030	-
2	0x0001	0x0010	0x0020	0x0030	-
1(Old)	0x0000	0x0010	0x0020	0x0030	-

F.5 Compatibility

The roadside system and OBE which are installed the updated basic API shall also install the first version of the basic API.

F.6 Selection of Version

F.6.1 Setting of version information

The procedure to set the basic API version information is as follows:

(1) The roadside system sets the version management table which identifies the basic API versions mounted in it.

(2) The roadside system makes valid local ports assigned to the basic API of all versions installed in it.

F.6.2 Selection procedure

The procedure to select the version of the basic API is as follows:

(1) The roadside system acquires from each API local port numbers whose connection is completed at initial connection, and creates the local port list for the OBE.

(2) The roadside system acquires local port number combinations specified by the latest version registered in its version management table, and compares them with the local port list for the OBE.

(3) When there is a same combination in the local port list for the OBE, the roadside system notifies each API of local port numbers to be used (Figure F.6-1).

(4) When there haven't a same combination in the local port list for the OBE, the roadside

system acquires local port number combinations specified by the next candidate version registered in its version management table, compares them with the local port list for the OBE, and searches for a same combination. The roadside system shall repeat the processing until a same combination is found.

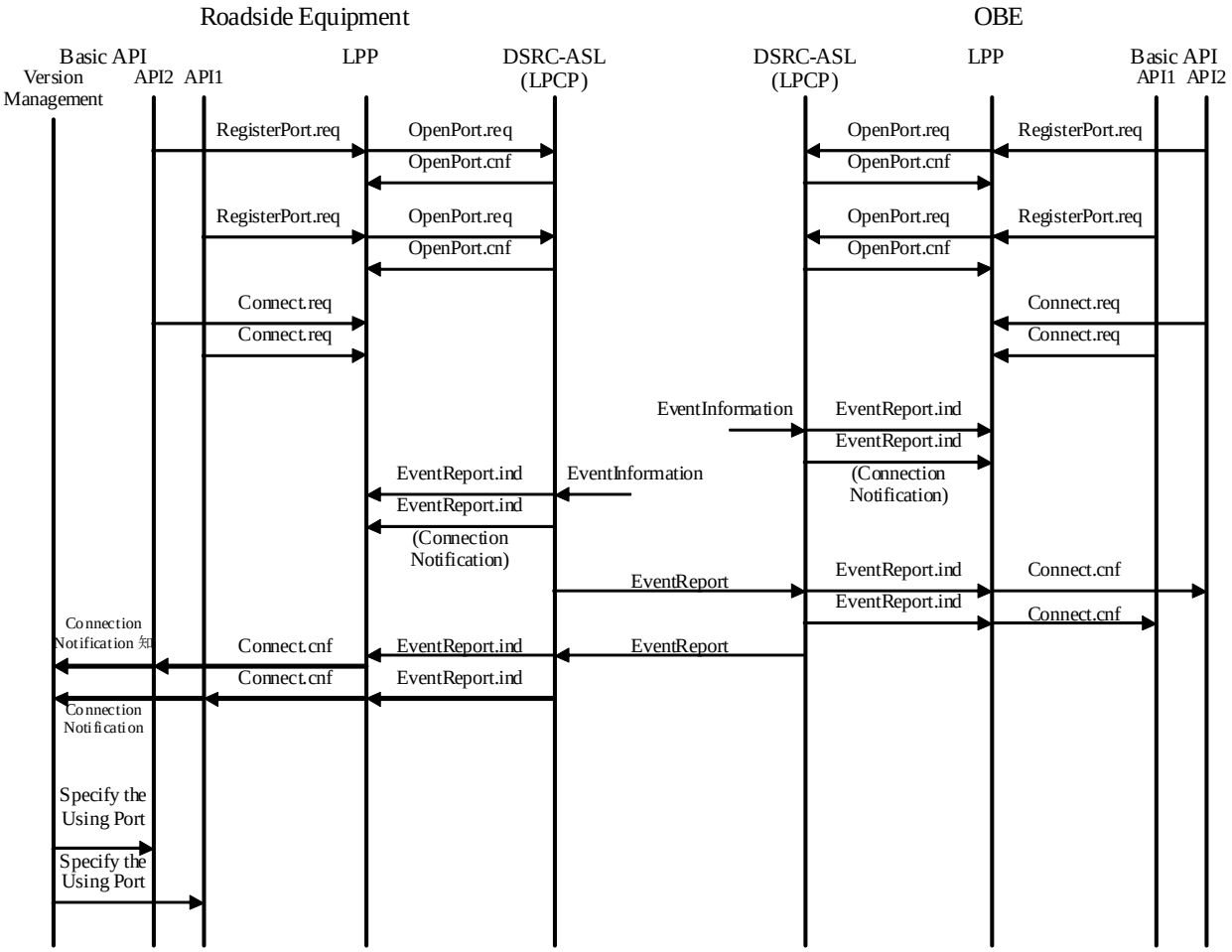


Figure F.6-1 Selection Procedure of Version

Annex G Push-type Information Delivery Application

G.1 Client Information Notification Command [Informative]

For client information notification command, it is possible to prescribe “supplementInfo” domain separately and use for individual service and future expansion.

A regulation regarding to “supplementInfo” is not a scope of this specification, but an example of items, which are stipulated by Standard of ITS On-Board Unit (JEITA TT6001 - 6004), are described in attached table G1-1 -1-2 for your reference.

Table G1-1 Format of supplementInfo

	7(MSB)	6	5	4	3	2	1	0(LSB)
1	Supplement Informatioon Code							
2 :	Supplement Information of each code							

Table G1-2 Supplement Information List

code	type	remarks
0	don’t use	
1	ITS On-Board Unit	OBE based on JEITA TT6001~6004 Supplement Information of this code is defined in TT6002
2~255	Reserved	Reserved For future use

G.2 Important Notice about Push-type Information Delivery Application

G.2.1 Important notice about number assignment of “Push ID”

Note for number assignment of “PushID” in road side system is shown below

(a) Common important notice about both point-to-point communication and broadcast communication

- As a general rule, same “PushID” should be used for identical contents*; different “PushID” should be used by contents.
- “PushID” may be managed as different systems by link address as a unit

(b) Important notice about broadcast communication

- In case of contents, which DC=0/1 is intermingled, are transmitted, it is recommended to use different “PushID” for type of DC flag

- In broadcast communication if plural contents are transmitted by repeating transmit system; do not carry out repeat transmission of more than 129 contents

- In broadcast communication if plural contents are transmitted by repeating transmit system; if new contents is transmitted with same “PushID”, “PushID” should be used after more than 128 of commands (DC=1) with other “PushID” are transmitted.

Note:Identical content means same type of contents and contents, which contain same data.

G.2.2 Other notices

- If massage to be conveyed is same and contents are different, transmit with proper judgment by road side as transmitting only one.

- It is recommended that repeating transmission system in broadcast communication is used when service is carried out by using application/contents type, which repetition checks are not carried out in executing application.

G.3 Operation Example of OBE triggered by RC/DC Flag

Table G3-1 Operation of the OBE triggered by the RC flag (point-to-point communication)

			Existence of cache data of same PushID			
			not there		there	
Event	PushOperation (Point-to-point communication)	RC=1	Transfer to application	Register to Cache	Transfer to application	Update Cache data
		RC=0	Transfer to application	-	Transfer to application	-

Table G3-2 Operation of the OBE triggered by the DC flag (broadcast communication)

			Existence of same PushID memory (for duplicate check)			
			There		Not there	
Event	PushOperation (Broadcast Communication)	DC=1	Transfer to application	Memory PushID*	Detect duplication	-
		DC=0	Transfer to application	-	Transfer to application	-
	Disconnect.ind		-	Abort all PushID		Abort all PushID

Note:Memorized “PushID” is 128 and the eldest “PushID” is deleted if more than 129 “PushID” are received

Amendment History

DSRC (Dedicated Short Range Communication) Basic Application Interfaces Specification

ITS FORUM GUIDELINE
(ITS FORUM RC-004)
The 1.1th edition amendment history

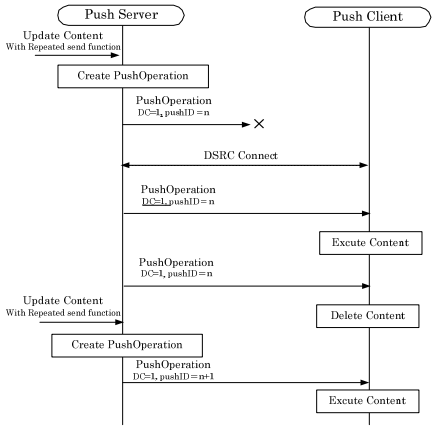
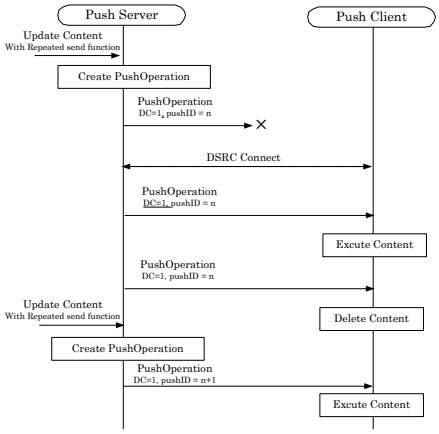
Amendment item1　　Amendment List about problem of PushID

Table Amendment1-1 Amendment list about problem of PushID(1/3)

<table>
<tr><th>Page</th><th>Para. no</th><th>Reason</th><th>Content of Amendment</th><th>Present</th><th>Page</th></tr>
<tr><td>120</td><td>3.4.2.2.1</td><td rowspan="3">Add “Duplicate Check” flag</td><td>Table3.4-1 the “PushOperation” command format.
<table><tr><td></td><td>7 (MSB)</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0 (LSB)</td></tr><tr><td>1</td><td colspan="4">commandTypepush(0)</td><td colspan="2">RES</td><td>RC</td><td>IS</td></tr><tr><td>2</td><td colspan="8">pushID</td></tr><tr><td></td><td colspan="8">:</td></tr></table></td><td>Table3.4-1 the “PushOperation” command format.
<table><tr><td></td><td>7(MSB)</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0(LSB)</td></tr><tr><td>1</td><td colspan="4">commandTypepush(0)</td><td>RES</td><td>DC</td><td>RC</td><td>IS</td></tr><tr><td>2</td><td colspan="8">pushID</td></tr><tr><td></td><td colspan="8">:</td></tr></table></td><td>130</td></tr>
<tr><td>120</td><td>3.4.2.2.1</td><td>(1) RC
RC stands for “RequireCashe”, and gives instruction to the push client to (or not to) hold content data. This field is set to “true (1)” for requesting holding, and stores “false (0)” for not requesting holding.</td><td>(3) DC
DC stands for “Duplicate Check”, and gives instruction to the push client to (or not to) execute the duplicate check. This field is set to “true (1)” for requesting the duplicate check, and is set to “false (0)” for not requesting the duplicate check.
This flag is valid only in broadcast communication, and is set to “0” in point-to-point communication.

(4) RC
RC stands for “RequireCache”, and gives instruction to the push client to (or not to) hold content data. This field is set to “true (1)” for requesting holding, and is set to “false (0)” for not requesting holding. This flag is valid only in point-to-point communication, and is set to “0” in broadcast communication.</td><td>130-131</td></tr>
<tr><td>132</td><td>3.4.3</td><td>PushOperation ::= SEQUENCE {
res BIT STRING(SIZE(2)),
requireCache BOOLEAN,
isSegment BOOLEAN,
pushId INTEGER(0..255),
applicationType ApplicationType,
contentType ContentType,
contentSize INTEGER(0..4294967295),
pushBody OCTET STRING
}</td><td>PushOperation ::= SEQUENCE {
res BIT STRING(SIZE(1)),
duplicateCheck BOOLEAN,
requireCache BOOLEAN,
isSegment BOOLEAN,
pushId INTEGER(0..255),
applicationType ApplicationType,
contentType ContentType,
contentSize INTEGER(0..4294967295),
pushBody OCTET STRING
}</td><td>145</td></tr>
</table>

Table Amendment1-1　Amendment list about problem of PushID of PushID(2/3)

Page	Para. no	Reason	Content of Amendment	Present	Page
140-141	3.4.5.2.3	Revided the data transfer procedure	3.4.5.2.3 Data transfer procedure in push type information delivery using broadcast communication (1) The push server creates a push operation commands "PushOperation", and distributes contents periodically. The same Push ID is used until it is updated. (2) When the mobile station enters the DSRC communication area, the push client receives "PushOperation" sent in the step (1). (3) The push client executes the processing for the received content in accordance with the content type and application type specified by "contentType" and "applicationType". (4) When receiving "PushOperation" and push ID received in (2) in the same communication area, the push client aborts the received "PushOperation". Reference: When a same push ID is sent consecutively using the LPP in the step (1), the replay function of the transaction is used. Accordingly, the duplicate received data in the step (4) is annuled by the LPP, and is not given to the push-type information delivery application. Figure 3.4-6 shows an example of the data transfer procedure sequence in push type broadcast delivery  Figure 3.4-6 example of the data transfer procedure sequence in push type information delivery using broadcast communication	3.4.5.2.3 Data transfer procedure in push type information delivery using broadcast communication (1) The push server creates one or more push operation commands "PushOperation" in accordance with a request from the service application, and distributes contents. At this time, if the service application does not specify the use of repeated send function, the push server sets "0" to the DC flag, and sends the created "PushOperation". If the service application specifies the use of repeated send function, the push server sets "1" to the DC flag, and sends the created "PushOpeartion" repeatedly.Refer to Attached data G2.1 (b) for consideration about push ID assignment in broadcast communication. (2) When the mobile station enters the DSRC communication area, the push client receives "PushOperation" sent in the step (1). The push client holds its push ID if the DC flag value is "1". (3) The push client executes the processing for the received content in accordance with the content type and application type specified by "contentType" and "applicationType". (4) When the push client receives "PushOperation" whose DC flag value is "1" and push ID remains held in the same communication area, the client aborts the received "PushOperation". Note: When the LPP sends the DSRC disconnection notice to the push client, the client shall annul the push ID stored if the DC flag value is "1". The push client can store up to 128 push IDs. If the push client receives 129 or more push IDs, it shall annul the push ID in turn from the one stored earliest. Reference: When a same push ID is sent consecutively using the LPP in the step (1), the replay function of the transaction is used. Accordingly, the duplicate received data in the step (4) is annuled by the LPP, and is not given to the push-type information delivery application. Figure 3.4-6(a) shows an example of the data transfer procedure sequence in push type broadcast delivery using the repeated send function, and Fig. 3.4-6(b) shows an example of the data transfer procedure sequence in push type delivery using broadcast communication without using the repeated send function.  Figure 3.4-6 (a) example of the data transfer procedure sequence in push type information delivery using broadcast communication (with repeated send function)	154-156

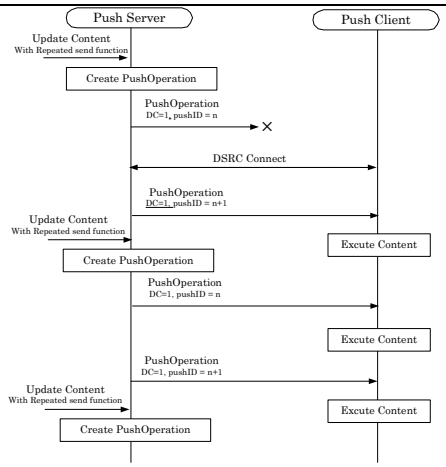


Figure 3.4-6 (b) example of the data transfer procedure sequence in push type information delivery using broadcast communication without the repeated send function

Table Amendment1-1 Amendment list about problem of PushID (3/3)

Page	Para. no	Reason	Content of Amendment	Present	Page
203	Annex C4	Add service examples		Add the following hedline to the service using Psuh-type information delivery in 224 page. (a) Point-to-point information providing service using the OBE ID Add the following services from 225 page to 227 page. (b) Point-to-point information providing service for many contents. (c) Broadcast information providing service using the repeated send function. (d) Broadcast information providing service using the dedicated application.	224-227
Add	Annex G2	Add the note for push type information broadcast application		Add Annex G2 "Note for push type information broadcast application".	252-253
Add	Annex G3	Add the operation example of OBE triggered by RC/DC flag		Add Annex G3 "Operation Example of OBE triggered by RC/DC Flag".	253

Amendment item2 Amendment List about Addition of Application Type / Content Type

Table Amendment 2-1 Amendment List about Addition of Application Type / Content Type (1/2)

Page	Para. no	Reason	Content of Amendment	Present	Page

132-133	3.4.3	Reviced the description of ASN.1	ApplicationType ::= CHOICE {
			:
			vics [8] NULL,
			text-display [9] NULL,
			other [10..254] NULL,
			private [255] OCTET STRING
			}
			
			ContentType ::= CHOICE{
			:
			text-x-html [7] NULL,
			otherTextType [8..15] NULL,
			:
			video-qt [51] NULL,
			reservedForFutureVideoType [52..63] NULL,
			:
			dsrc-mime [131] NULL,
			otherType [132..239] NULL,
			private [240..255] NULL
			}
				ApplicationType ::= CHOICE {
				:
				vics [8] NULL,
				text-display [9] NULL,
				safety [10] NULL,
				image-display [11] NULL,
				other [12..254] NULL,
				private [255] OCTET STRING
				}
				
				ContentType ::= CHOICE{
				:
				text-x-html [7] NULL,
				text-tts [8] NULL
				otherTextType [9..15] NULL,
				:
				video-qt [51] NULL,
				video-wmv [52] NULL,
				reservedForFutureVideoType [53..63] NULL,
				:
				dsrc-mime [131] NULL,
				dsrc-safety [132] NULL,
				dsrc-multipart [133] NULL,
				otherType [134..239] NULL,
				private [240..255] NULL
				}
					145-147

Table Amendment 2-1 Amendment List about Addition of Application Type / Content Type (2/2)

<table>
<tr><th>Page</th><th>Para. no</th><th>Reason</th><th>Content of Amendment</th><th>Present</th><th>Page</th></tr>
<tr><td>149</td><td>3.4.6</td><td>Add theapplication type</td><td>
Table 3.4-14 Application type and content type list
<table>
<tr><th>Application</th><th>Idnetifier</th><th>Value</th><th>Remarks</th></tr>
<tr><td colspan="4">:</td></tr>
<tr><td>Sound Player</td><td>sound-player</td><td>0x03</td><td>Sound content type</td></tr>
<tr><td>Movie Player</td><td>video-player</td><td>0x04</td><td>Video content type</td></tr>
<tr><td>TTS</td><td>tts</td><td>0x05</td><td>Generally text/plain content type is specified. ---</td></tr>
<tr><td colspan="4">:</td></tr>
<tr><td>Text display</td><td>text-display</td><td>0x09</td><td>Display text data.</td></tr>
<tr><td>Others</td><td>others</td><td>0x0C-0xFE</td><td>(Note2)</td></tr>
<tr><td>Any</td><td>private</td><td>0xFF</td><td>specify kind of application by any text</td></tr>
</table>
</td><td>
Table 3.4-14 Application type list
<table>
<tr><th>Application</th><th>Identifier</th><th>Value</th><th>Remarks</th></tr>
<tr><td colspan="4">:</td></tr>
<tr><td>Sound Player</td><td>sound-player</td><td>0x03</td><td></td></tr>
<tr><td>Movie Player</td><td>video-player</td><td>0x04</td><td></td></tr>
<tr><td>TTS</td><td>tts</td><td>0x05</td><td></td></tr>
<tr><td colspan="4">:</td></tr>
<tr><td>Text display</td><td>text-display</td><td>0x09</td><td></td></tr>
<tr><td><u>Driving Safety Support</u></td><td><u>safety</u></td><td><u>0x0A</u></td><td></td></tr>
<tr><td><u>Image Display</u></td><td><u>image-display</u></td><td><u>0x0B</u></td><td></td></tr>
<tr><td>Others</td><td>others</td><td>0x0C-0xFE</td><td>(Note2)</td></tr>
<tr><td>Any</td><td>private</td><td>0xFF</td><td></td></tr>
</table>
</td><td>164</td></tr>
</table>

149-151	3.4.6	Add the conent type	Table 3.4-15 Content type list (see table below)	Table 3.4-15 Content type and content type list (see table below)	165-166

Table 3.4-15 Content type list

Contents Type	Value	Format of pushBody	Remarks
:			
text/x-html	0x07	X-HTML file	X-HTML text
otherTextType	0x08-0x0F	-	text type(Note 2)
image/*	0x10	Image file	Any Image Type. Specified kind of image by OCTET STRING (Note)
:			
video/qt	0x33		QuickTime file
otherVideoType	0x34-0x3F		For video type(Note2)
message/*	0x40	Mail message defined RFC822	Any message type. Specified kind of message by OCTET STRING (Note)
otherMessage Type	0x41-0x4F	-	For message type (Note)
:			
multipart/*	0x60	Multi-partm essage	Any multipart type. Specified kind of multipart message type by OCTET STRING (Note)
otherMultipart Type	0x61-0x7F	-	For multipart type (Note2)
:			
dsrc/mime	0x83	MIME encoded text file	MIME encoded data
otherType	0x84-0xEF		(Note 2)
private	0xF0-FF		For private use (available for any usage)

Table 3.4-15 Content type and content type list

Contents Type	Value	Format of pushBody	Remarks
:			
text/x-html	0x07	X-HTML file	X-HTML text
text/tts	0x08	TTS file	JEITA TT-6004
otherTextType	0x09-0x0F	-	(Note2)
image/*	0x10	Image file	(Note1)
:			
video/qt	0x33		QuickTime file
video/wmv	0x34		WMV file
otherVideoType	0x35-0x3F		(Note2)
message/*	0x40	Mail message defined RFC822	(Note1)
otherMessageType	0x41-0x4F	-	(Note2)
:			
multipart/*	0x60	Multi-part message	(Note1)
otherMultipartType	0x61-0x7F	-	(Note2)
:			
dsrc/mime	0x83	MIME encoded text file	
dsrc/safety	0x84		
dsrc/multipart	0x85	Multi-part contents format	Refer to JEITA TT6003
otherType	0x86-0xEF		(Note 2)
private	0xF0-FF		

Amendment item3 Amendment List about Client Information Notice Command

Table Amendment 3-1 Amendment List about Client Information Notice Command

Page	Para. no	Reason	Content of Amendment	Present	Page
Add	Annex G1	Add the description about client information notice command		G1 Client Information Notification Command [Informative] For client information notification command, it is possible to prescribe "supplementInfo" domain separately and use for individual service and future expansion. A regulation regarding to "supplementInfo" is not a scope of this specification, but an example of items, which are stipulated by Standard of ITS On-vehicle device (JEITA TT6001 - 6004), are described in attached table G1-1 – 1-2 for your reference. (see tables below)	252

Table G1-1 Format of supplementInfo

	7(MSB)	6	5	4	3	2	1	0 (LSB)
1	Supplement Information Code							
2 :	Supplement Information of each code							

Table G1-2 Supplement Information List

Code	Type	Remarks
0	don't	

				use		
				1	ITS On-Board Unit	OBE based on JEITA TT6001~6004. Supplement Information of this code isdefined in TT6002.
				2~255	Reserved	Reserved For future use

Amendment item4 Amendment List about Others

Table Amendment 4-1 Amendment List about Others

Page	Para. no	Reason	Content of Amendment	Present	Page
3	1.5	Additional Revision		Add Standard Specifications "JEITA ITS On-Board Unit".	4-5

22
77
109
169
181
	3.1.3
	3.2.3
	3.3.3
	3.5.3
	3.6.3
		Correct a misdescription	ObuDenialResponse ::=SEQUENCE{
			status INTEGER(0..255),
			supplementInfo OCTET STRING(SIZE(0..127))
			}
				ObuDenialResponse ::=SEQUENCE{
				status INTEGER(0..255),
				supplementInfo OCTET STRING(SIZE(0..255))
				}
					26
					85
					118
					189
					202

135	3.4.3	Correct a misdescription	PushAbortOperation ::= SEQUENCE {
			:
			supplementInfo OCTET STRING(SIZE(0..127))
			}
				PushAbortOperation ::= SEQUENCE {
				:
				supplementInfo OCTET STRING(SIZE(0..255))
				}
					148

136	3.4.3	Correct a misdescription	ClientInformation ::= SEQUENCE {
			:
			supplementInfo OCTET STRING(SIZE(0..127))
			}
				ClientInformation ::= SEQUENCE {
				:
				supplementInfo OCTET STRING(SIZE(0..255))
				}
					149

120	3.4.2.2.1	Correct a misdescription	(3) RC RC stands for "RequireCashe", and gives instruction to the push client to (or not to) hold content data.	(4) RC RC stands for "RequireCache", and gives instruction to the push client to (or not to) hold content data.	131
122	3.4.2.2.2	Correct a misdescription	(3) RC RC stands for "RequireCashe", and gives instruction to the push client to (or not to) hold content data.	(3) RC RC stands for "RequireCache", and gives instruction to the push client to (or not to) hold content data.	132

[BLANK]

Amendment History
DSRC (Dedicated Short Range Communication) Basic Application Interfaces Specification

ITS FORUM GUIDELINE
(ITS FORUM RC-004)
The 1.2th edition amendment history

Amendment item1　　List about Addition of Reference

Table Amendment1-1 List about Addition of Reference

Page	Para. no	Reason	Content of Amendment	Present	Page
6	1.5	Addition of Reference	1.5 References This Guideline incorporates provisions from other publications by dated or undated reference. These normative references are cited at the appropriate places in the text and the publications are listed hereafter. For dated references, subsequent amendments to or revisions of any of these publications apply to this Guideline only when incorporated in it by amendment or revision. For undated references the latest edition of the publication referred to applies. ARIB STD-T75 DEDICATED SHORT-RANGE COMMUNICATION (DSRC) SYSTEM : JEITA TT-6004 Speech synthesizer symbols for ITS on-Board Unit	1.5 References This Guideline incorporates provisions from other publications by dated or undated reference. These normative references are cited at the appropriate places in the text and the publications are listed hereafter. For dated references, subsequent amendments to or revisions of any of these publications apply to this Guideline only when incorporated in it by amendment or revision. For undated references the latest edition of the publication referred to applies. ARIB STD-T75 DEDICATED SHORT-RANGE COMMUNICATION (DSRC) SYSTEM : JEITA TT-6004 Speech synthesizer symbols for ITS on-Board Unit ISO15628:2007 Road transport and traffic telematics -- Dedicated short range communication (DSRC) -- DSRC application layer ISO24103:2009 Intelligent transport systems -- Communications access for land mobiles (CALM) -- Media adapted interface layer (MAIL) ISO29281:2011 Intelligent transport systems -- Communications access for land mobiles (CALM) -- Non-IP networking	4

Amendment item2　　List about Correction of Misdescription

Table Amendment2-1 List about Correction of Misdescription

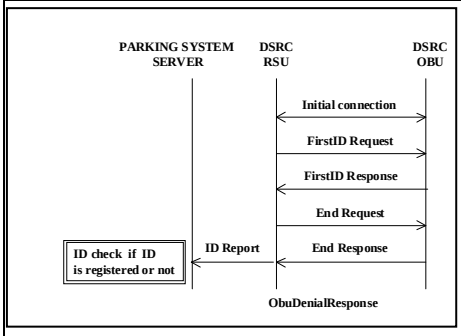
<table>
<tr><th>Page</th><th>Para. no</th><th>Reason</th><th>Content of Amendment</th><th>Present</th><th>Page</th></tr>
<tr><td>112</td><td>3.3.2</td><td rowspan="9">Correction of Misdescription</td><td>Table3.3-10 Committed Information Acquisition Response Command Format
<table><tr><td></td><td>7 (MSB)</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0 (LSB)</td></tr><tr><td>1</td><td colspan="4">Version(1)</td><td colspan="5">fill</td></tr><tr><td>2</td><td colspan="9">:</td></tr></table></td><td>Table3.3-10 Committed Information Acquisition Response Command Format
<table><tr><td></td><td>7 (MSB)</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0 (LSB)</td></tr><tr><td>1</td><td colspan="3">version</td><td colspan="6">fill</td></tr><tr><td>2</td><td colspan="9">:</td></tr></table></td><td>109</td></tr>
<tr><td>218</td><td>Annex C C.1</td><td> 

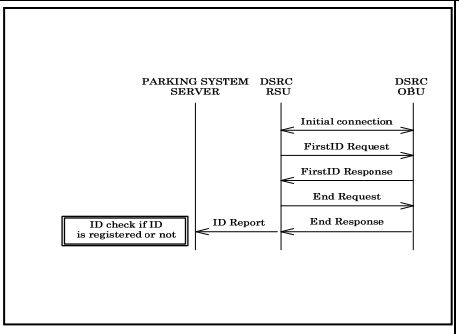
Figure C1-1 Communication transaction example (for plain text) in the settlement processing system using the OBE ID application interface</td><td> 

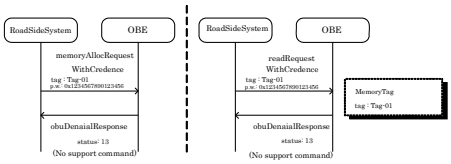
Figure C1-1 Communication transaction example (for plain text) in the settlement processing system using the OBE ID application interface</td><td>216</td></tr>
<tr><td>233</td><td>Annex D D.2.1</td><td>(1) When receiving “memoryallocRequest” or “memoryFreeRequest” from the roadside system, the OBE shall send the OBU denial response command “obuDenialResponse” whose status is “13: No support command" to the roadside system.</td><td>(1) When receiving “memoryallocRequest” or “memoryFreeRequest” from the roadside system, the OBE shall send the OBU denial response command “obuDenialResponse” whose status is “12: No support command" to the roadside system.</td><td>231</td></tr>
<tr><td>234</td><td>Annex D D.2.2</td><td>(2) When receiving “memoryAllocRequestWithCredence”, “memoryFreeRequestWithCredence”, “readRequestWithCredence”, “writeRequestWithCredence”, "readBulkRequestWithCredence" or “writeBulkRequestWithCredence” from the roadside system, the OBE shall send the OBE denial response command “obuDenialResponse” whose status is “13: No support command" to the roadside system.</td><td>(2) When receiving “memoryAllocRequestWithCredence”, “memoryFreeRequestWithCredence”, “readRequestWithCredence”, “writeRequestWithCredence”, "readBulkRequestWithCredence" or “writeBulkRequestWithCredence” from the roadside system, the OBE shall send the OBE denial response command “obuDenialResponse” whose status is “12: No support command" to the roadside system.</td><td>232</td></tr>
<tr><td>234</td><td>Annex D</td><td> 

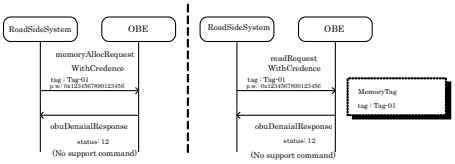
Figure D2-1 Example1 of Password Command</td><td> 

Figure D2-1 Example1 of Password Command</td><td>232</td></tr>
<tr><td>237</td><td>Annex D D.3</td><td>For memory tags for which the SPF is essential, the OBE accepts only commands by way of the LP3, and sends the OBE denial response command "obuDenialResponse” whose status is “10: SPF violation” to the roadside system for commands by way of the LP2.</td><td>For memory tags for which the SPF is essential, the OBE accepts only commands by way of the LP3, and sends the OBE denial response command "obuDenialResponse” whose status is “11: SPF violation” to the roadside system for commands by way of the LP2.</td><td>235</td></tr>
</table>

Amendment item3　Amendment List about Addition of Contactless IC Smart Card-compliance

Table Amendment3-1 Amendment List about Addition of Contactless IC Smart Card-compliance

<table>
<tr><th>Page</th><th>Para. no</th><th>Reason</th><th>Content of Amendment</th><th>Present</th><th>Page</th></tr>
<tr><td>11</td><td>2.3.3</td><td>Addition of Contactless IC Smart Card-compliance</td><td>2.3.3 IC card access application
IC card access application provides the function to access the IC card on request from the system on the roadside by the method prescribed by ISO/IEC7816. The function of the IC card access application is exclusively applicable to the ISO/IEC7816-compliant IC card.</td><td>2.3.3 IC card access application
IC card access application provides the function to access the IC card on request from the system on the roadside by the method prescribed by ISO/IEC7816. The function of the IC card access application is exclusively applicable to the ISO/IEC7816-compliant IC card and contactless IC smart card.</td><td>8</td></tr>
<tr><td>13</td><td>2.4</td><td></td><td>Table 2.4-1 Local port number of basic primitive application functions (example)
<table>
<tr><th>Port number</th><th>Basic primitive application functions</th><th>Data flow</th></tr>
<tr><td>0x0C00</td><td>OBE ID communication</td><td rowspan="2">Road to vehicle</td></tr>
<tr><td>0x0C01 - 0x0C07</td><td>For future use</td></tr>
<tr><td>0x0C08</td><td>OBE basic instruction</td><td rowspan="4">Road to vehicle</td></tr>
<tr><td>0x0C09</td><td>OBE instruction response</td></tr>
<tr><td>0x0C0A</td><td>Push-type information delivery</td></tr>
<tr><td>0x0C0B - 0x0C0F</td><td>For future use</td></tr>
<tr><td>0x0C10</td><td>IC card access</td><td rowspan="2">Road to vehicle /Vehicle to road, Use of card</td></tr>
<tr><td>0x0C11 - 0x0C17</td><td>For future use</td></tr>
<tr><td>0x0C18</td><td>OBE memory access</td><td rowspan="2">Road to vehicle /Vehicle to road, Use of memory</td></tr>
<tr><td>0x0C19 - 0x0C1F</td><td>For future use</td></tr>
</table></td><td>Table 2.4-1 Local port number of basic primitive application functions (example)
<table>
<tr><th>Port number</th><th>Basic primitive application functions</th><th>Data flow</th></tr>
<tr><td>0x0C00</td><td>OBE ID communication</td><td rowspan="2">Road to vehicle</td></tr>
<tr><td>0x0C01 - 0x0C07</td><td>For future use</td></tr>
<tr><td>0x0C08</td><td>OBE basic instruction</td><td rowspan="4">Road to vehicle</td></tr>
<tr><td>0x0C09</td><td>OBE instruction response</td></tr>
<tr><td>0x0C0A</td><td>Push-type information delivery</td></tr>
<tr><td>0x0C0B - 0x0C0F</td><td>For future use</td></tr>
<tr><td>0x0C10</td><td>IC card access</td><td rowspan="3">Road to vehicle /Vehicle to road, Use of card</td></tr>
<tr><td>0x0C11</td><td>Contactless IC smart card access</td></tr>
<tr><td>0x0C12 - 0x0C17</td><td>For future use</td></tr>
<tr><td>0x0C18</td><td>OBE memory access</td><td rowspan="2">Road to vehicle /Vehicle to road, Use of memory</td></tr>
<tr><td>0x0C19 - 0x0C1F</td><td>For future use</td></tr>
</table></td><td>10</td></tr>
<tr><td>116</td><td>3.3.4</td><td></td><td>Table 3.3-11 Relationship with other DSRC related standards
<table>
<tr><th></th><th>DSRC related standard</th><th>Content using in this application</th></tr>
<tr><td>1</td><td>DSRC Standard</td><td>AID=18 (DSRC Application Sub Layer)</td></tr>
<tr><td>2</td><td>NCP of ASL</td><td>LPCP (Local Port Control Protocol)</td></tr>
<tr><td>3</td><td>Port number of LPCP</td><td>0x0C10</td></tr>
<tr><td>4</td><td>Transaction Service</td><td>The relationships of the other DSRC related standards using this application are shown in below.</td></tr>
</table></td><td>Table 3.3-11 Relationship with other DSRC related standards
<table>
<tr><th></th><th>DSRC related standard</th><th>Content using in this application</th></tr>
<tr><td>1</td><td>DSRC Standard</td><td>AID=18 (DSRC Application Sub Layer)</td></tr>
<tr><td>2</td><td>NCP of ASL</td><td>LPCP (Local Port Control Protocol)</td></tr>
<tr><td>3</td><td>Port number of LPCP</td><td>Contact IC: 0x0C10
Contactless IC: 0x0C11</td></tr>
<tr><td>4</td><td>Transaction Service</td><td>The relationships of the other DSRC related standards using this application are shown in below.</td></tr>
</table></td><td>113</td></tr>
<tr><td>216</td><td>Annex B B.2</td><td></td><td>TableB2-1 Local Port Number of Basic Application
<table>
<tr><th>Normal Port</th><th>Secure Port</th><th>Application</th><th>Remarks</th></tr>
<tr><td>0x0C00</td><td>0x0C20</td><td>OBE ID Communication Application</td><td rowspan="2"><data flow>
RSU<-OBU</td></tr>
<tr><td>0x0C01-0x0C07</td><td>0x0C21-0x0C27</td><td>Reserved For future use</td></tr>
<tr><td>0x0C08</td><td>0x0C28</td><td>OBE Basic Indication Application</td><td rowspan="4"><data flow>
RSU->OBU</td></tr>
<tr><td>0x0C09</td><td>0x0C29</td><td>OBE Instruction Response Application</td></tr>
<tr><td>0x0C0A</td><td>0x0C2A</td><td>Push-type information delivery application</td></tr>
<tr><td>0x0C0B-0x0C0F</td><td>0x0C2B-0x0C2F</td><td>Reserved For future use</td></tr>
<tr><td>0x0C10</td><td>0x0C30</td><td>ICCard Access Application</td><td rowspan="3"><data flow>
RSU<->OBU using IC Card</td></tr>
<tr><td>0x0C11-0x0C17</td><td>0x0C31-0x0C37</td><td>Reserved For future use</td></tr>
<tr><td>0x0C18</td><td>0x0C38</td><td>OBE Memory Access Application</td><td rowspan="3"><data flow>
RSU<->OBU using Memory</td></tr>
<tr><td>0x0C19-0x0C1F</td><td>0x0C39-0x0C3E</td><td>Reserved For future use</td></tr>
<tr><td>-</td><td>0x0C3F</td><td>Security Platform Management Entity</td></tr>
</table></td><td>TableB2-1 Local Port Number of Basic Application
<table>
<tr><th>Normal Port</th><th>Secure Port</th><th>Application</th><th>Remarks</th></tr>
<tr><td>0x0C00</td><td>0x0C20</td><td>OBE ID Communication Application</td><td rowspan="2"><data flow>
RSU<-OBU</td></tr>
<tr><td>0x0C01-0x0C07</td><td>0x0C21-0x0C27</td><td>Reserved For future use</td></tr>
<tr><td>0x0C08</td><td>0x0C28</td><td>OBE Basic Indication Application</td><td rowspan="4"><data flow>
RSU->OBU</td></tr>
<tr><td>0x0C09</td><td>0x0C29</td><td>OBE Instruction Response Application</td></tr>
<tr><td>0x0C0A</td><td>0x0C2A</td><td>Push-type information delivery application</td></tr>
<tr><td>0x0C0B-0x0C0F</td><td>0x0C2B-0x0C2F</td><td>Reserved For future use</td></tr>
<tr><td>0x0C10</td><td>0x0C30</td><td>ICCard Access Application</td><td rowspan="3"><data flow>
RSU<->OBU using IC Card</td></tr>
<tr><td>0x0c11</td><td>0x0C31</td><td>Contactless IC Smart Card Access Application</td></tr>
<tr><td>0x0C12-0x0C17</td><td>0x0C32-0x0C37</td><td>Reserved For future use</td></tr>
<tr><td>0x0C18</td><td>0x0C38</td><td>OBE Memory Access Application</td><td rowspan="3"><data flow>
RSU<->OBU using Memory</td></tr>
<tr><td>0x0C19-0x0C1F</td><td>0x0C39-0x0C3E</td><td>Reserved For future use</td></tr>
<tr><td>-</td><td>0x0C3F</td><td>Security Platform Management Entity</td></tr>
</table></td><td>214</td></tr>
</table>

Amendment item4　Amendment List about Addition of Application Type and Content Type

Table Amendment4-1 Amendment List about Addition of Application Type and Content Type

<table>
<tr><th>Page</th><th>Para. no</th><th>Reason</th><th>Content of Amendment</th><th>Present</th><th>Page</th></tr>
<tr><td>140
-141</td><td>3.4.3</td><td>Addition of Application Type</td><td>ApplicationType ::= CHOICE {
default [0] NULL,
browser [1] NULL,
mailer [2] NULL,
sound-player [3] NULL,
video-player [4] NULL,
tts [5] NULL,
mobile-device-browser [6] NULL,
store [7] NULL,
vics [8] NULL,
text-display [9] NULL,
safety [10] NULL,
image-display [11] NULL,
other [12-254] NULL,
private [255] OCTET STRING
}</td><td>ApplicationType ::= CHOICE {
default [0] NULL,
browser [1] NULL,
mailer [2] NULL,
sound-player [3] NULL,
video-player [4] NULL,
tts [5] NULL,
mobile-device-browser [6] NULL,
store [7] NULL,
vics [8] NULL,
text-display [9] NULL,
safety [10] NULL,
image-display [11] NULL,
payment [12] NULL,
other [13-254] NULL,
private [255] OCTET STRING
}</td><td>137
-138</td></tr>
<tr><td>141
-142</td><td>3.4.3</td><td>Addition of Content Type</td><td>ContentType ::= CHOICE{
everyType [0] OCTET STRING,
text [1] OCTET STRING,
: :
: :
dsrc-safety [132] NULL,
dsrc-multipart [133] NULL,
otherType [134..239] NULL,
private [240..255] NULL
}</td><td>ContentType ::= CHOICE{
everyType [0] OCTET STRING,
text [1] OCTET STRING,
: :
: :
dsrc-safety [132] NULL,
dsrc-multipart [133] NULL,
dsrc-privateSpot_text_plain [134] NULL,
dsrc-privateSpot_image_jpeg [135] NULL,
dsrc-privateSpot_image_gif [136] NULL,
dsrc-privateSpot_image_bmp [137] NULL,
dsrc-privateSpot_image_tiff [138] NULL,
dsrc-privateSpot_image_png [139] NULL,
otherType [140..239] NULL,
private [240..255] NULL
}</td><td>138
-139</td></tr>
<tr><td>161</td><td>3.4.6</td><td>Addition of Application Type and Correction of Misdescription</td><td>Table 3.4-14 Application type list
<table><tr><th>Application</th><th>Identifier</th><th>Value</th><th>Remarks</th></tr><tr><td>default</td><td>default</td><td>0x00</td><td></td></tr><tr><td>Web Brouser</td><td>browser</td><td>0x01</td><td></td></tr><tr><td>Mailer</td><td>mailer</td><td>0x02</td><td></td></tr><tr><td>Sound Player</td><td>sound-player</td><td>0x03</td><td></td></tr><tr><td>Movie Player</td><td>video-player</td><td>0x04</td><td></td></tr><tr><td>TTS</td><td>tts</td><td>0x05</td><td></td></tr><tr><td>Mobile Brouser</td><td>mobile-device-browser</td><td>0x06</td><td></td></tr><tr><td>Store</td><td>store</td><td>0x07</td><td></td></tr><tr><td>VICS</td><td>vics</td><td>0x08</td><td></td></tr><tr><td>Text display</td><td>text-display</td><td>0x09</td><td></td></tr><tr><td>Driving Safety Support</td><td>safety</td><td>0x0A</td><td></td></tr><tr><td>Image Display</td><td>image-display</td><td>0x0B</td><td></td></tr><tr><td>Others</td><td>others</td><td>0x0C-0xFE</td><td>(Note 2)</td></tr><tr><td>Any</td><td>private</td><td>0xFF</td><td></td></tr></table></td><td>Table 3.4-14 Application type list
<table><tr><th>Application</th><th>Identifier</th><th>Value</th><th>Remarks</th></tr><tr><td>default</td><td>default</td><td>0x00</td><td></td></tr><tr><td>Web Browser</td><td>browser</td><td>0x01</td><td></td></tr><tr><td>Mailer</td><td>mailer</td><td>0x02</td><td></td></tr><tr><td>Sound Player</td><td>sound-player</td><td>0x03</td><td></td></tr><tr><td>Movie Player</td><td>video-player</td><td>0x04</td><td></td></tr><tr><td>TTS</td><td>tts</td><td>0x05</td><td></td></tr><tr><td>Mobile Browser</td><td>mobile-device-browser</td><td>0x06</td><td></td></tr><tr><td>Store</td><td>store</td><td>0x07</td><td></td></tr><tr><td>VICS</td><td>vics</td><td>0x08</td><td></td></tr><tr><td>Text display</td><td>text-display</td><td>0x09</td><td></td></tr><tr><td>Driving Safety Support</td><td>safety</td><td>0x0A</td><td></td></tr><tr><td>Image Display</td><td>image-display</td><td>0x0B</td><td></td></tr><tr><td>Payment</td><td>payment</td><td>0x0C</td><td></td></tr><tr><td>Others</td><td>others</td><td>0x0D-0xFE</td><td></td></tr><tr><td>Any</td><td>private</td><td>0xFF</td><td></td></tr></table></td><td>158</td></tr>
<tr><td>162
-163</td><td>3.4.6</td><td>Addition of Content Type</td><td>Table 3.4-15 Content type list
<table><tr><th>Contents Type</th><th>Value</th><th>fprmat of pushBody</th><th>Remarks</th></tr><tr><td>*/*</td><td>0x00</td><td></td><td>(Note 1)</td></tr><tr><td>text/*</td><td>0x01</td><td></td><td>(Note 1)</td></tr><tr><td>:</td><td>:</td><td>:</td><td>:</td></tr><tr><td>:</td><td>:</td><td>:</td><td>:</td></tr><tr><td>dsrc/safety</td><td>0x84</td><td></td><td></td></tr><tr><td>dsrc/multipart</td><td>0x85</td><td>multipart contents format</td><td>Refer to JEITA TT6003</td></tr><tr><td>otherType</td><td>0x86-0xEF</td><td></td><td>(Note 2)</td></tr><tr><td>private</td><td>0xF0-FF</td><td></td><td></td></tr></table></td><td>Table 3.4-15 Content type list
<table><tr><th>Contents Type</th><th>Value</th><th>fprmat of pushBody</th><th>Remarks</th></tr><tr><td>*/*</td><td>0x00</td><td></td><td>(Note 1)</td></tr><tr><td>text/*</td><td>0x01</td><td></td><td>(Note 1)</td></tr><tr><td>:</td><td>:</td><td>:</td><td>:</td></tr><tr><td>:</td><td>:</td><td>:</td><td>:</td></tr><tr><td>dsrc/safety</td><td>0x84</td><td></td><td></td></tr><tr><td>dsrc/multipart</td><td>0x85</td><td>multipart contents format</td><td>Refer to JEITA TT6003</td></tr><tr><td>dsrc/privateSpot_text_plain</td><td>0x86</td><td>Shift JIS line feed code CR+LF</td><td></td></tr></table></td><td>159
-160</td></tr>
</table>

				dsrc/privateSpot_text_plain	0x86	Shift JIS line feed code CR+LF	Refer to Guideline for In-vehicle Display Systems	
				dsrc/privateSpot_image_jpeg	0x87	Image File Binary Format		
				dsrc/privateSpot_image_gif	0x88			
				dsrc/privateSpot_image_bmp	0x89			
				dsrc/privateSpot_image_tiff	0x8A			
				dsrc/privateSpot_image_png	0x8B			
				otherType	0x8C-0xEF		(Note 2)	
				private	0xF0-FF			

[BLANK]