CPA System API Specification

cpa-support

Version 0.0.3

Revision history . . . 1
1. Introduction . . . 3
2. Overview . . . 4
3. Transactions . . . 8
3.1. REST API . . . 8
3.1.1. General description . . . 8
3.1.2. Interaction . . . 8
3.2. SMPP Protocol . . . 9
3.2.1. General description . . . 9
3.2.2. Interaction . . . 9
3.3. Interaction assumptions . . . 10
3.3.1. Indirect interaction with subscriber . . . 10
3.3.2. Charging . . . 10
3.4. Transaction messages . . . 11
3.4.1. Mobile terminated message . . . 11
Description . . . 11
Supported protocols and formats . . . 11
Processing flow . . . 11
3.4.2. Mobile originated message . . . 12
Description . . . 12
Supported protocols and formats . . . 12
Processing flow . . . 12
3.4.3. Delivery report . . . 13
Description . . . 13
Receive DLR . . . 13
Get DLR . . . 14
4. Syntax notation . . . 15
4.1. REST API . . . 15
4.1.1. JSON . . . 15
Send MT . . . 15
Receive MO . . . 18
Receive DLR . . . 20
Get DLR . . . 22
4.1.2. XML . . . 24
Send MT . . . 24
Receive MO . . . 27
Receive DLR . . . 29
Get DLR . . . 31
4.2. SMPP Protocol . . . 33
4.2.1. Send MT . . . 33
4.2.2. Receive MO . . . 34

4.2.3. Receive DLR . . . 35
5. Parameters definition . . . 36
5.1. REST API . . . 36
5.1.1. Header parameters . . . 36
Authentication and authorization . . . 36
Content-Type . . . 36
5.1.2. Mandatory parameters . . . 37
date . . . 37
destination / sin . . . 38
content / body . . . 38
mid . . . 39
nodeId . . . 39
service . . . 39
source / sn . . . 40
status . . . 40
report.status . . . 41
report.status.errorCode . . . 42
5.1.3. Optional parameters . . . 43
Restriction . . . 43
bearerType / bearer . . . 43
callbackNum . . . 44
contentType / content-type . . . 44
eos . . . 45
errorId . . . 46
errorMsg / error . . . 46
report.status.error . . . 46
rid . . . 47
serviceType / paid . . . 47
5.2. SMPP Protocol . . . 48
5.2.1. Header parameters . . . 48
5.2.2. Mandatory parameters . . . 49
command_status . . . 49
data_coding . . . 49
destination_addr . . . 49
esm_class . . . 49
message_id . . . 50
message_state . . . 50
service_type . . . 50
sm_length . . . 50
source_addr . . . 50
5.2.3. Optional parameters . . . 51

message_payload . . . 51
receipted_message_id . . . 51
short_message . . . 51
6. Message types . . . 53
6.1. SMS message . . . 53
6.1.1. SMS / plain . . . 54
6.1.2. SMS / silent . . . 55
6.1.3. SMS / binary . . . 56
6.2. USSD message . . . 57
6.2.1. USSD / push . . . 57
6.2.2. USSD / menu . . . 58
6.3. Message encoding . . . 61
6.3.1. Encoding for text/plain content type . . . 61
Encoding for DLR, MO messages with text/plain content type . . . 61
Encoding for MT message with text/plain content type . . . 61
6.3.2. Encoding for smpp/binary content type . . . 61
Encoding for MO message with smpp/binary content type . . . 61
Encoding for MT message with smpp/binary content type . . . 62
7. General . . . 63
7.1. HTTP verbs . . . 63
7.2. HTTP status codes . . . 63
7.3. Error information . . . 64
JSON error types . . . 64
XML error types . . . 65
SMPP error types . . . 67
7.4. Data encoding scheme . . . 68
7.5. Glossary . . . 69

Revision history

Issue	Description	Author
V. 0.0.1 16.08.2018	1. Specification creation. Basic set of chapters and sections: *1. Introduction* *2. Overview* *2.1. Header description* *3. Transactions* *3.1. Interaction general description* *3.2. Mobile terminated message* *3.3. Delivery report* *4. General* *4.1. HTTP verbs* *4.2. HTTP status codes* *4.3. Error information* *4.4. Glossary*	Tetyana Tiunova / Softengi
V. 0.0.2 13.11.2018	1. Added paragraph Support and development to *Chapter 1. Introduction* 2. Added section 3.3. Mobile originated message to *Chapter 3. Transactions* 3. Added Chapter 4. Parameters definition 4. Added Chapter 5. Message types 5. Added section 6.4. Data encoding scheme to *Chapter 6. General* 6. Amended Chapter 2. Overview: - added POST method example to Mobile originated message 7. section Header parameters moved to *Chapter 4. Parameters definition* 8. Amended section 3.2. Mobile terminated message: - added `rid` parameter description - added `eos` parameter description - refactored tables structure 9. Amended section 3.4. Delivery report: - refactored tables structure 10. Amended section JSON error types: - added new error types: 1007, 1056, 1057, 1058, 1059, 1060, 1901 - removed deprecated error types: 1044, 1046, 1047, 1048, 1049, 1050, 1052, 1055 11. Amended section XML error types: - added new error type: 101 12. Amended section 6.5. Glossary - added new key terms 13. Changed chapters and sections numbering	Tetyana Tiunova / Softengi

Issue	Description	Author
V. 0.0.3 31.12.2018	1. Refactored Chapter 3. Transactions: - added section 3.1. REST API - added section 3.2. SMPP Protocol - added section 3.4. Transaction messages - section *Interaction parties* renamed to Interaction assumptions 2. Added Chapter 4. Syntax notation 3. Amended Chapter 5. Parameters definition: - added section 5.2. SMPP Protocol - amended section 5.1.2. REST API: Mandatory parameters: added length recommendations to parameter `content / body` - amended section 5.1.3. REST API: Optional parameters: added `callbackNum` parameter description 4. Amended paragraph Purpose and Scope: - SMPP Protocol Description added to document scope 5. Amended Chapter 2. Overview: - added note that examples of syntax notation are provided for REST API only - removed *header parameters* form *Data parameters short description* 6. Changed resources for MO, MT, DLR via REST API 7. Renamed transaction to receive DLR 8. Amended Chapter 6. Message types: - added note that examples of syntax notation are provided for REST API only 9. Amended section 7.3. Error information: - added new error types: 1014, 1062, 1063, 1064 to subsection JSON error types - added subsection SMPP error types 10. Amended section 7.5. Glossary: - added new key terms - changed description for terms: *Message Identifier*, *Request Identifier* 11. Changed chapters and sections numbering	Tetyana Tiunova / Softengi

Chapter 1. Introduction

This document describes APIs of Content Provider Access System.

Confidentiality Statement

The information in this document is designated as 'Commercial in Confidence'. Therefore, the document or the information in it should not be provided to any party other than Kyivstar JSC (KS) partners.

Purpose and Scope

The purpose of this document is to describe interfaces, that enables interaction between software of Content Provider Access System and Content Providers.
The following is included:

- Basic system description
- REST API description for `JSON` and `XML` data-interchange formats
- SMPP protocol description
- Supported message types
- Syntax notation

Support and development

Content Provider Access System (CPA System) is designed to provide high performance and low latency of message exchange. CPA System implements REST API and SMPP protocol to communicate with Content Provider software. By now CPA System REST API allows to use both `JSON` and `XML` data-interchange formats for interaction. `XML` format is implemented to allow legacy compatibility with Content Providers software, which had been integrated with previous version of CPA System. Support of `XML` format is considered as unpromising direction and expires on **01.04.2019**.

Chapter 2. Overview

The Content Provider Access System (CPA System) is a system that enables interaction between subscriber and Content Provider. By Content Provider should be basically meant a system that provides any informational services to subscriber. By subscriber should be basically meant a person that employs the services provided by a telecommunication system or information processing systems which transfers information.

Interaction between subscriber and Content Provider derives from the following main needs:

1. Subscriber wants to order defined informational services.
2. Content Provider wants to provide informational services to subscriber.

Detailing above mentioned interaction next examples may be considered.

Example 1. Mobile originated message

Subscriber wants to order access to movie via short message.
A schematic **representation of Example 1** is depicted in the following sketch.

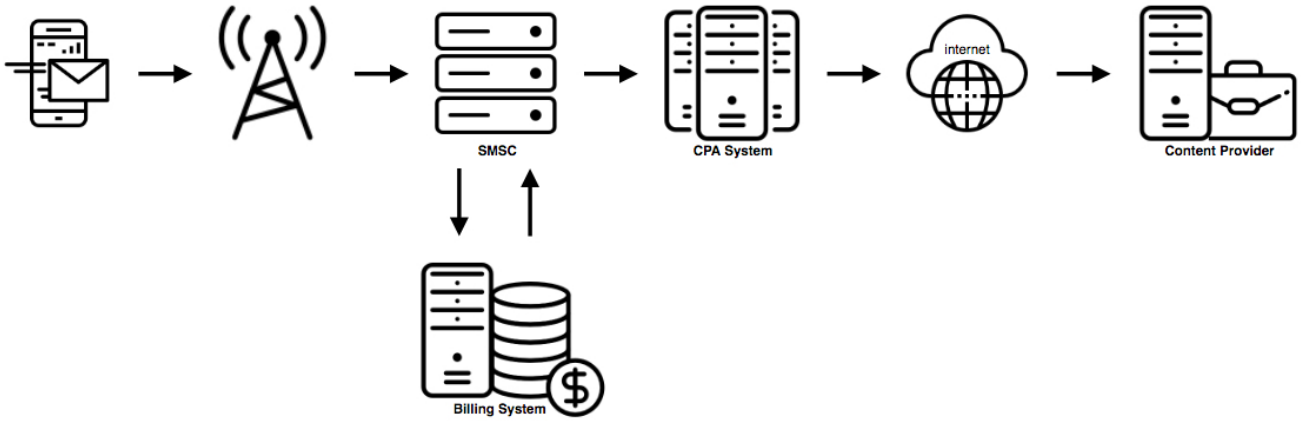

Figure 1. Mobile originated message

As depicted in the **Figure 1** subscriber sends a short message containing a defined token to a short number dedicated for movie ordering. External content delivery system (SMS Centre) receives a short message, performs required processing with billing system and forwards a short message to CPA System. CPA System receives subscriber's short message, processes this message, transforms it into request and forwards it to Content Provider. Content Provider receives request and provides access to movie for subscriber.

Mentioned example defines that subscriber himself requests an informational service. In current document is used term **'Mobile originated message' (MO)** to define message with request for informational service from subscriber to Content Provider.

Examples of syntax notation are provided **for REST API only**.
Considering that CPA System REST API allows to use both `JSON` and `XML` data-interchange formats, examples are provided for both formats.

For deeper detailing of **Example 1** is provided **POST method example of CPA System REST API**, which CPA System uses in order to send message to Content Provider.

JSON request

```
POST http://{environment}/api/requests

Authorization: Basic dm9yZGVsOnZvcmRlbA==
Content-Type: application/json

{
    "rid": "d8549def-5d93-4e52-8b63-b282ccefa971",
    "date": "1513296000000",
    "source": "380672240001",
    "destination": "101999",
    "bearerType": "sms",
    "contentType": "text/plain",
    "content": "Hello Provider!"
}
```

Data parameters short description

`rid` - subscriber's request identifier
`date` - date when CPA System received subscriber's request
`source` - subscribers MSISDN (phone number)
`destination` - service number
`bearerType` - message type
`contentType` - type of content that is being sent to Provider
`content` - content that is being sent to Provider

XML request

```
POST http://{environment}/cpa2/receiver

Authorization: Basic dm9yZGVsOnZvcmRlbA==
Content-Type: application/xml

<message rid="d8549def-5d93-4e52-8b63-b282ccefa971" bearer="SMS">
    <sn>101999</sn>
    <sin>380670000001</sin>
    <service>content-request</service>
    <status date="Wed, 31 Jan 20018 16:20:00 GMT"/>
    <body content-type="text/plain">Hello Provider!</body>
</message>
```

Data parameters short description

`rid` - subscriber's request identifier
`bearer` - message type
`sn` - service number

`sin` - subscribers MSISDN (phone number)
`service` - message type
`date` - date when request was sent to Content Provider
`content-type` - type of content that is being sent to Provider
`body` - content that is being sent to Provider

Example 2. Mobile terminated message

Content Provider wants to provide an advertisement short message to subscriber.
A schematic **representation of Example 2** is depicted in the following sketch.

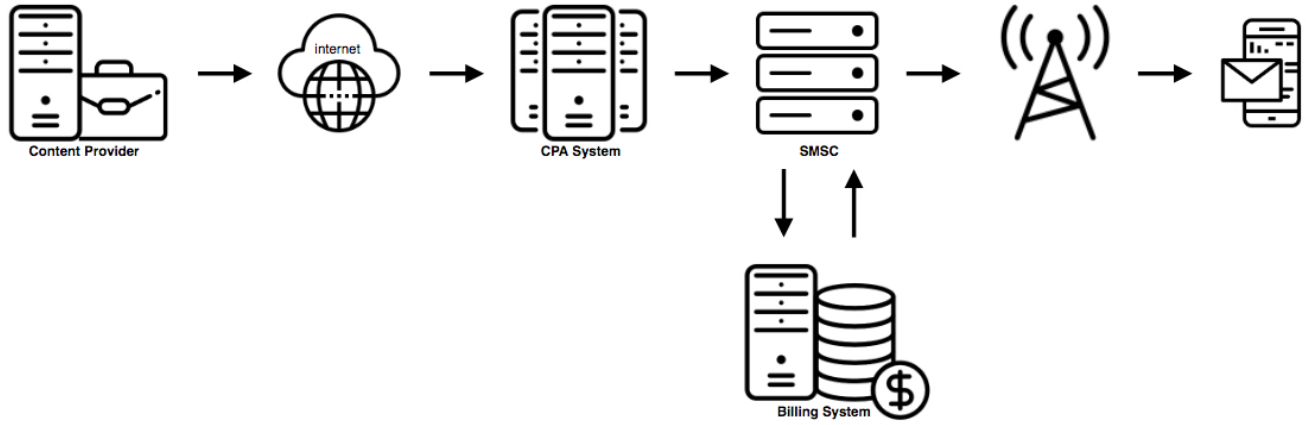

Figure 2. Mobile terminated message

As depicted in the **Figure 2** Content Provider sends request to CPA System using CPA System API. CPA System in its turn processes this request, transforms it into message and forwards it to External content delivery system (SMS Centre). SMS Centre receives message, performs required processing with billing system, and forwards a short message to subscriber.

Mentioned example defines that Content Provider sends an advertisement short message without prior subscriber's request. In current document is used term **'Mobile terminated message' (MT)** to define message with informational service from Content Provider to subscriber.

For deeper detailing of **Example 2** is provided **POST method example** of CPA System REST API, which Content Provider uses in order to send short message to subscriber.

JSON request

```
POST http://{environment}/api/contents

Authorization: Basic dm9yZGVsOnZvcmRlbA==
Content-Type: application/json

{
    "source": "101999",
    "destination": "380670000001",
    "serviceType": "true"
    "bearerType": "sms"
    "contentType": "text/plain"
    "content": "Hello World!"
}
```

Data parameters short description

`source` - service number
`destination` - subscribers MSISDN (phone number)
`serviceType` - type of payment for the message
`bearerType` - message type
`contentType` - type of content that is being sent to subscriber
`content` - content that is being sent to subscriber

XML request

```
POST http://{environment}/cpa2/receiver

Authorization: Basic dm9yZGVsOnZvcmRlbA==
Content-Type: application/xml

<message mid="904705f0-3c5e-4556-8339-81b7fd2e3d83" paid="false" bearer="SMS">
    <sn>101999</sn>
    <sin>380670000001</sin>
    <body content-type="text/plain">Hello World!</body>
</message>
```

Data parameters short description

`mid` - message identifier
`paid` - type of payment for the message
`bearer` - message type
`sn` - service number
`sin` - subscribers MSISDN (phone number)
`content-type` - type of content that is being sent to subscriber
`body` - content that is being sent to subscriber

Chapter 3. Transactions

The Content Provider Access System implements following APIs for interaction between CPA System and Content Provider:

- **REST API**
- **SMPP Protocol**

Used for:

- accepting content request from subscriber and forwarding it to Content Provider (Mobile originated message)
- accepting content response from Content Provider and forwarding it to subscriber (Mobile terminated message)
- accepting request from Content Provider for retrieving Delivery report (Delivery report)
- sending Delivery report to Content Provider (Delivery report).

3.1. REST API

3.1.1. General description

CPA System REST API is designed as a messaging protocol and implements next methods:

`POST`

- for accepting content response from Content Provider
- for forwarding subscriber's content request to Content Provider
- for sending Delivery report to Content Provider

`GET`

- for accepting request from Content Provider for retrieving Delivery report

CPA System REST API supports next data-interchange formats:

- `JSON`
- `XML`

3.1.2. Interaction

Interaction means an act of sending one message. The party which is sending the message is called *the sender* and the party which receives the message is called *the receiver*. To complete the interaction, *sender* and *receiver* perform the following steps:

1. *The sender* generates the message.
2. *The sender* initiates an HTTP connection to the URL corresponding to *the receiver*. (How the sender learned which URL to use, is outside the scope of this document – one could safely assume that these URLs are exchanged before, as a part of CPA System and Content Provider

initial setup.)

3. *The sender* sets appropriate authorization information in request, according to used authorization scheme.
4. *The sender* submits request, transmitting the message in the body of the request.
5. *The receiver* validates message (performs internal required checks)
    1. If message passes validation, *the receiver* generates HTTP response, which depends on initial message format:
        - for `JSON` - response has HTTP status code of 202 and contains message identifier (`mid`);
        - for `XML` - response has HTTP status code of 200 and contains processing status result.
    2. If message does not pass validation, *the receiver* generates HTTP response, which contains HTTP status and error message (error message varies depending on initial message format).
6. *The receiver* sends HTTP response to *the sender*.

Invalid message is not proceed further.

3.2. SMPP Protocol

3.2.1. General description

CPA System enables interaction via SMPP Protocol as closely as possible to Short Message Peer to Peer Protocol Specification v3.4.
CPA System implements next protocol data units (PDUs):

`SUBMIT_SM`

- for accepting content response from Content Provider

`SUBMIT_SM_RESP`

- for informing Content Provider on content response acceptance

`DELIVER_SM`

- for forwarding subscriber's content request to Content Provider
- for sending Delivery report to Content Provider

`DELIVER_SM_RESP`

- for informing CPA System on subscriber's content request or Delivery report acceptance.

3.2.2. Interaction

Interaction means a process of request/response exchange. Within interaction, CPA System and Content Provider take following roles according to SMPP Protocol Specification v3.4:

- CPA System functions as *Short Message Service Centre (SMSC)*
- Content Provider functions as *External Short Message Entity (ESME)*

CPA System SMPP Protocol implementation allows to exchange messages in both directions, so both CPA System and Content Provider should function as **transceivers**. Interaction should be performed as a typical SMPP request/response sequence.

3.3. Interaction assumptions

3.3.1. Indirect interaction with subscriber

CPA System APIs enable interaction between CPA System and Content Provider in order to either transport subscriber's order for content to Content Provider or provide content to subscriber. However neither CPA System nor Content Provider does not interact with subscriber directly.

Mobile originated content request from subscriber is sent through defined communication channel to External content delivery system, which in it's turn delivers it to CPA System and than CPA System forwards it to Content Provider. The same way Mobile terminated content response from Content Provider is sent to CPA System, than forwarded to External content delivery system, which in it's turn delivers it to subscriber through defined communication channel.

For the purpose of current document can be used phrases like 'deliver/send to subscriber', 'receive/accept from subscriber' in order to facilitate understanding of general business logic. Upon further thought it is necessary to understand that 'deliver/send' to- or 'receive/accept' from subscriber means indirect interaction through External content delivery system.

3.3.2. Charging

Content provided to subscriber can require subscriber's charging. However CPA System does not charge subscriber by itself. In case if charges for content are required, CPA System can only initiate pre-check whether subscriber's core balance is enough to be charged. Mentioned pre-check operation is performed by external system.

For the purpose of current document can be used phrases like 'to initiate subscriber's core balance pre-check', 'to initiate subscriber's charging' or common in order to facilitate understanding of general business logic. Upon further thought it is necessary to understand that any actions for subscriber's charging are performed by external system.

3.4. Transaction messages

3.4.1. Mobile terminated message

Description

MT message - a message generated by Content Provider and sent to CPA System in order to provide informational services to subscriber.

Supported protocols and formats

In order to accept MT message from Content Provider CPA System uses:

via REST API:	**Send MT POST** method for `JSON` and `XML` formats
via SMPP Protocol:	`SUBMIT_SM` PDU

Processing flow

In order to accept MT message from Content Provider, CPA System performs following steps:

1. CPA System accepts MT from Content Provider.
2. CPA System performs all required MT checks. Required checks can include but are not limited to the following:
   - defining subscriber's billing system type (prepaid or postpaid);
   - pre-check whether subscriber's core balance is enough to be charged.
     1. If at least one of required checks passed unsuccessfully and MT cannot be proceed further, CPA System sends error message to Content Provider and does not proceed MT.
     2. If required checks passed successfully, CPA System continues processing MT.
3. CPA System generates *message identifier* (optional).
4. CPA System stores MT to database.
5. CPA System defines appropriate External content delivery system and forwards MT.
6. When acceptance response from External content delivery system is received, CPA System informs Content Provider on successful MT acceptance.

3.4.2. Mobile originated message

Description

MO message - a message generated by subscriber to order informational services form Content Provider.

Supported protocols and formats

In order to send MO message to Content Provider CPA System uses:

via REST API:	**Receive MO POST** method for `JSON` and `XML` formats
via SMPP Protocol:	`DELIVER_SM` PDU

Processing flow

In order to send MO message to Content Provider, CPA System performs following steps:

1. accepts MO from External Content Delivery System;
2. generates *request identifier*;
3. sends MO to Content Provider;
4. stores MO in database.

3.4.3. Delivery report

Description

Delivery report - a message generated by CPA System and sent to Content Provider in order to inform on result of sending content to subscriber. Initially, sending of Delivery report is performed by CPA System automatically, but it also can be retrieved by Content Provider manually.

Successful Delivery report

Successful Delivery report scenario considers that MT message passed the whole processing flow on CPA System side and was forwarded to External content delivery system successfully. Further External content delivery system informs CPA System that content was sent to subscriber.

Unsuccessful Delivery report

Unsuccessful Delivery report scenario considers that forwarding MT message to External content delivery system passed unsuccessfully. In such case CPA System sends Unsuccessful Delivery report to Content Provider.

Receive DLR

Supported protocols and formats

In order to send Delivery report to Content Provider CPA System uses:

via REST API:	**Receive DLR POST** method for `JSON` and `XML` formats
via SMPP Protocol:	`DELIVER_SM` PDU

Processing flow

1. CPA System accepts message on content delivery from External content delivery system.
2. CPA System transforms message on content delivery into DLR.
3. CPA System sends DLR to Content Provider. In case if Content Provider is unavailable, CPA System tries to send DLR via all defined Content Provider's urls.
4. CPA System stores DLR to database in order it can be retrieved later.

Get DLR

Supported protocols and formats

Retrieving Delivery report is implemented only via REST API and varies depending on data-interchange format

In order to retrieve Delivery report from CPA System, Content Provider uses:

via REST API:	**Get DLR GET** method for `JSON` format **Get DLR POST** method for `XML` format

Processing flow

In order to retrieve Delivery report from CPA System, Content Provider performs following steps depending on data-interchange format.

`JSON`

1. Content Provider sends request to CPA System.
2. CPA System retrieves DLR from database:
   - if DLR is found in database, than CPA System answers to Content Provider with HTTP code 200 and body containing DLR;
   - if DLR is not found in database, than CPA System answers to Content Provider with HTTP code 404.

In case if Content Provider is unavailable, CPA System tries to send DLR via all defined Content Provider's urls.

`XML`

1. Content Provider sends request to CPA System.
2. CPA System:
   1. accepts request from Content Provider;
   2. checks DLR in database:
      - if DLR is found in database, than CPA System answers to Content Provider on request acceptance with HTTP code 200; retrieves DLR from database; tries to send DLR via all defined Content Provider's urls;
      - if DLR is not found in database, than CPA System answers to Content Provider with HTTP code 200 and error body.

In case if Content Provider is unavailable, CPA System tries to send DLR via all defined Content Provider's urls.

Chapter 4. Syntax notation

4.1. REST API

In current section are provided detailed syntax notation and examples of CPA System REST API.

4.1.1. JSON

Send MT

Resource

```
http://{environment}/api/contents
```

Method

```
POST
```

MT HTTP request body minimal required set example

```
{
    "source": "101999",
    "destination": "380670000001",
    "content": "Hello World!"
}
```

MT HTTP request body max set example

```
{
    "rid": "d8549def-5d93-4e52-8b63-b282ccefa971",
    "source": "101999",
    "destination": "380670000001",
    "serviceType": "true",
    "callbackNum": "69",
    "bearerType": "sms",
    "contentType": "text/plain",
    "content": "Hello World!"
}
```

MT HTTP request body data parameters short description

`rid` - subscriber's request identifier
`source` - service number
`destination` - subscribers MSISDN (phone number)
`serviceType` - type of payment for the message
`callbackNum` - Content Provider's callback number
`bearerType` - message type

`contentType` - type of content that is being sent to subscriber
`content` - content that is being sent to subscriber

Success HTTP response example

```
Status code: 202

{
    "mid": "904705f0-3c5e-4556-81b7fd2e3d83"
}
```

Error HTTP response example

```
Status code: 400

{
    "mid": "904705f0-3c5e-4556-81b7fd2e3d83",
    "errorId": 1019,
    "errorMsg": "Invalid destination address"
}
```

MT HTTP response body data parameters short description

`mid` - message identifier
`errorId` - error identifier
`errorMsg` - error message

Table 1. MT HTTP request body data parameters detailed description

#	Name	Type	Required	Description	Ref
1	`rid`	string	no	subscriber's request identifier (UUID)	rid
2	`source`	string [3-12]	yes	service number	source / sn
3	`destination`	string [12-32]	yes	subscribers MSISDN (phone number) or hashed value	destination / sin
4	`serviceType`	string	no	type of payment for the message	serviceType / paid
5	`callbackNum`	string	no	Content Provider's callback number	callbackNum
6	`bearerType`	string	no	message type	bearerType / bearer
7	`contentType`	string	no	type of content in field 'content' that is being sent to subscriber	contentType / content-type
8	`content`	string	yes	content that is being sent to subscriber	content / body

Table 2. MT HTTP response body data parameters detailed description

#	Name	Type	Required	Description	Ref
1	`mid`	string	yes	message identifier (UUID)	mid
2	`errorId`	integer	no	error identifier	errorId
3	`errorMsg`	string	no	error message	errorMsg / error

Receive MO

Resource Example

```
http://{environment}/api/requests
```

Method

```
POST
```

MO HTTP request body example

```
{
    "rid": "d8549def-5d93-4e52-8b63-b282ccefa971",
    "date": "1513296000000",
    "source": "380672240001",
    "destination": "101999",
    "bearerType": "sms",
    "contentType": "text/plain",
    "content": "Hello Provider!"
}
```

MO HTTP request body data parameters short description

`rid` - subscriber's request identifier
`date` - date when CPA System received subscriber's request
`source` - subscribers MSISDN (phone number)
`destination` - service number
`bearerType` - message type
`contentType` - type of content that is being sent to Provider
`content` - content that is being sent to Provider

Success HTTP response example

```
Status code: 204
```

Table 3. MO HTTP request body data parameters detailed description

#	Name	Type	Required	Description	Ref
1	`rid`	string	yes	subscriber's request identifier (UUID)	rid
2	`date`	integer	yes	date when CPA System received subscriber's request	date
3	`source`	string [12-32]	yes	subscribers MSISDN (phone number) or hashed value	source / sn
4	`destination`	string [3-12]	yes	service number	destination / sin
5	`bearerType`	string	no	message type	bearerType / bearer
6	`contentType`	string	no	type of content in field 'content' that is being sent to Content Provider	contentType / content-type
7	`content`	string	yes	content that is being sent to Content Provider	content / body

Receive DLR

Resource Example

```
http://{environment}/api/reports
```

Method

```
POST
```

DLR HTTP request body example (content delivered successfully)

```
{
    "mid": "904705f0-3c5e-4556-8339-81b7fd2e3d83",
    "date": "1513296000000",
    "status": "delivered"
}
```

DLR HTTP request body example (DLR expired)

```
{
    "mid": "904705f0-3c5e-4556-8339-81b7fd2e3d83",
    "date": "1513296000000",
    "status": "expired"
}
```

DLR HTTP request body data parameters short description

`mid` - message identifier
`date` - date when CPA System received Delivery report
`status` - Delivery report status
`errorId` - error identifier
`errorMsg` - error message

Success HTTP response example

```
Status code: 200

{
    "mid": "904705f0-3c5e-4556-8339-81b7fd2e3d83"
}
```

DLR HTTP response body data parameters short description

`mid` - message identifier

Table 4. DLR HTTP request body data parameters detailed description

#	Name	Type	Required	Description	Ref
1	`mid`	string	yes	message identifier (UUID)	mid
2	`date`	integer	no	date when CPA System received Delivery report	date
2	`status`	string	yes	Delivery report status	status
3	`errorId`	integer	no	error identifier	errorId
4	`errorMsg`	string	no	error message	errorMsg / error

Table 5. DLR HTTP response body data parameters detailed description

#	Name	Type	Required	Description	Ref
1	`mid`	string	yes	message identifier (UUID)	mid

Get DLR

Resource

```
http://{environment}/api/reports/{mid}
```

Method

```
GET
```

DLR HTTP request example

```
http://{environment}/api/reports/d8549def-5d93-4e52-8b63-b282ccefa971
```

DLR HTTP request uri parameters short description

`{mid}` - message identifier

Success HTTP response example

```
Status code: 200

{
    "mid": "904705f0-3c5e-4556-8339-81b7fd2e3d83",
    "date": "1513296000000",
    "status": "delivered"
}
```

Error HTTP response example

```
Status code: 404

{
    "mid": "904705f0-3c5e-4556-8339-81b7fd2e3d83",
    "errorId": 1040,
    "errorMsg": "Resource not found"
}
```

DLR HTTP response body data parameters short description

`mid` - message identifier
`date` - date when CPA System received Delivery report
`status` - delivery report status
`errorId` - error identifier
`errorMsg` - error message

Table 6. DLR HTTP request uri parameters detailed description

#	Name	Type	Required	Description	Ref
1	`mid`	string	yes	message identifier (UUID)	mid

Table 7. DLR HTTP response body data parameters detailed description

#	Name	Type	Required	Description	Ref
1	`mid`	string	yes	message identifier (UUID)	mid
2	`date`	integer	yes	date when CPA System received Delivery report	date
3	`status`	string	yes	Delivery report status	status
4	`errorId`	integer	no	error identifier	errorId
5	`errorMsg`	string	no	error message	errorMsg / error

4.1.2. XML

Send MT

Resource

```
http://{environment}/cpa2/receiver
```

Method

```
POST
```

MT HTTP request body minimal required set example

```
<message mid="904705f0-3c5e-4556-8339-81b7fd2e3d83">
    <sn>101999</sn>
    <sin>380670000001</sin>
    <body>Hello World!</body>
</message>
```

MT HTTP request body max set example

```
<message rid="d8549def-5d93-4e52-8b63-b282ccefa971"
    mid="904705f0-3c5e-4556-8339-81b7fd2e3d83" paid="false" bearer="SMS">
    <sn>101999</sn>
    <sin>380670000001</sin>
    <body content-type="text/plain">Hello World!</body>
</message>
```

MT HTTP request body data parameters short description

`rid` - subscriber's request identifier
`mid` - message identifier
`paid` - type of payment for the message
`bearer` - message type
`sn` - service number
`sin` - subscribers MSISDN (phone number)
`content-type` - type of content that is being sent to subscriber
`body` - content that is being sent to subscriber

Success HTTP response example

```
Status code: 200

    <report>
        <status errorCode="0">Accepted</status>
    </report>
```

Error HTTP response example

```
Status code: 200

    <report>
        <status error="Subscriber error" errorCode="23">Rejected</status>
    </report>
```

MT HTTP response body data parameters short description

`status` - processing status result
`error` - error message
`errorCode` - error identifier

Table 8. MT HTTP request data parameters detailed description

#	Tag name	Attribute name	Type	Required	Description	Ref
1	`message`	`rid`	string	no	subscriber's request identifier (UUID)	rid
2		`mid`	string	yes	message identifier (UUID)	mid
3		`paid`	string	no	type of payment for the message	serviceType / paid
4		`bearer`	string	no	message type	bearerType / bearer
5		`eos`	string	no	interaction termination identifier	eos
6	`sn`	-	string [3-12]	yes	service number	source / sn
7	`sin`	-	string [12-32]	yes	subscribers MSISDN (phone number) or hashed value	destination / sin
8	`body`	`content-type`	string	no	type of content in field 'content' that is being sent to subscriber	contentType / content-type
9		-	string	no	content that is being sent to subscriber	content / body

Table 9. MT HTTP response data parameters detailed description

#	Tag name	Attribute name	Type	Required	Description	Ref
1	`status`	`error`	string	no	error message	report.status.error
2		`errorCode`	string	yes	error identifier	report.status.errorCode
3		-	string	yes	request acceptance status	report.status

Receive MO

Resource Example

```
http://{environment}/api/requests
```

Method

```
POST
```

MO HTTP request body example

```
<message rid="d8549def-5d93-4e52-8b63-b282ccefa971" bearer="SMS">
    <sn>101999</sn>
    <sin>380670000001</sin>
    <service>content-request</service>
    <status date="Wed, 31 Jan 20018 16:20:00 GMT"/>
    <body content-type="text/plain">Hello Provider!</body>
</message>
```

MO HTTP request body data parameters short description

`rid` - subscriber's request identifier
`bearer` - message type
`sn` - service number
`sin` - subscribers MSISDN (phone number)
`service` - message type
`date` - date when CPA System received subscriber's request
`content-type` - type of content that is being sent to Provider
`body` - content that is being sent to Provider

Success HTTP response example

```
Status code: 200

<report>
    <status>Accepted</status>
</report>
```

Table 10. MO HTTP request body data parameters detailed description

#	Tag name	Attribute name	Type	Required	Description	Ref
1	`message`	`rid`	string	yes	subscriber's request identifier (UUID)	rid
2		`bearer`	string	yes	message type	bearerType / bearer
3	`sn`	-	string [3-12]	yes	service number	source / sn
4	`sin`	-	string [12-32]	yes	subscribers MSISDN (phone number) or hashed value	destination / sin
5	`service`	-	string	yes	defines request type: DLR or MO	service
6	`status`	`date`	string	yes	date when CPA System received subscriber's request	date
8	`body`	`content-type`	string	yes	type of content in field 'content' that is being sent to Content Provider	contentType / content-type
9		-	string	yes	content that is being sent to Content Provider	content / body

Table 11. MO HTTP response body data parameters detailed description

#	Tag name	Attribute name	Type	Required	Description	Ref
1	`status`	-	string	yes	request acceptance status	report.status

Receive DLR

Resource Example

```
http://{environment}/cpa2/receiver
```

Method

```
POST
```

DLR HTTP request body example (content delivered successfully)

```
<message mid="904705f0-3c5e-4556-8339-81b7fd2e3d83" nodeId="0">
    <service>delivery-report</service>
    <status date="Wed, 31 Jan 20018 16:20:00 GMT">Delivered</status>
</message>
```

DLR HTTP request body example (content not delivered)

```
<message mid="904705f0-3c5e-4556-8339-81b7fd2e3d83" nodeId="0">
    <service>delivery-report</service>
    <status date="Wed, 31 Jan 20018 16:20:00 GMT"
    error="SMSC rejected. Code 255">Undeliverable</status>
</message>
```

DLR HTTP request body data parameters short description

`mid` - message identifier
`nodeId` - CPA System node identifier
`service` - message type
`status` - Delivery report status
`date` - date when CPA System received Delivery report
`error` - error message (can optionally contain identifier)

Success HTTP response example

```
Status code: 200

<report>
    <status>Accepted</status>
</report>
```

DLR HTTP response body data parameters short description

`status` - result of SEND Delivery report request acceptance

Table 12. DLR HTTP request body data parameters detailed description

#	Tag name	Attribute name	Type	Required	Description	Ref
1	`message`	`mid`	string	yes	message identifier (UUID)	mid
2		`nodeId`	integer	yes	CPA System's node identifier	nodeId
3	`service`	-	string	yes	defines request type: DLR or MO	service
4	`status`	`date`	string	yes	date when CPA System received Delivery report	date
5		`error`	string	no	error message	errorMsg / error
6		-	string	yes	Delivery report status	status

Table 13. DLR HTTP response body data parameters detailed description

#	Tag name	Attribute name	Type	Required	Description	Ref
1	`status`	-	string	yes	request acceptance status	report.status

Get DLR

Annotation

`XML` format usage assumes that CPA System synchronously responses to Content Provider on result of *GET Delivery report* request acceptance and only later sends Delivery report itself.

In current section are described:

- *GET DLR* request `XML` structure;
- response on acceptance of *GET DLR* request.

DLR itself is provided in section Receive DLR

Resource

```
http://{environment}/cpa2/receiver
```

Method

```
POST
```

DLR HTTP request example

```
<message mid="904705f0-3c5e-4556-8339-81b7fd2e3d83">
    <service>delivery-request</service>
</message
```

DLR HTTP request body data parameters short description

`mid` - message identifier
`service` - message type

Success HTTP response example

```
Status code: 200

<report>
    <status errorCode="0">Accepted</status>
</report>
```

Error HTTP response example

```
Status code: 200

<report>
   <status error="Other unexpected new error" errorCode="34">Rejected</status>
</report>
```

DLR HTTP response body data parameters short description

`status` - result of GET Delivery report request acceptance
`error` - error message
`errorCode` - error identifier

Table 14. DLR HTTP request body data parameters detailed description

#	Tag name	Attribute name	Type	Required	Description	Ref
1	`message`	`mid`	string	yes	message identifier (UUID)	mid
2	`service`	-	string	yes	defines request type: DLR or MO	service

Table 15. DLR HTTP response body data parameters detailed description

#	Tag name	Attribute name	Type	Required	Description	Ref
1	`status`	`error`	string	no	error message	report.status.error
2		`errorCode`	string	yes	error identifier	report.status.errorCode
3		-	string	yes	request acceptance status	report.status

4.2. SMPP Protocol

In current section is provided detailed syntax notation and examples of CPA System SMPP Protocol implementation.

4.2.1. Send MT

SMPP PDU to send MT message to CPA System

```
submit_sm
```

SMPP PDU to inform Content Provider on submit_sm PDU acceptance

```
submit_sm_resp
```

Table 16. submit_sm PDU parameters detailed description

#	Name	Type	Required	Description	Ref
1	`source_addr`	c-octet string	yes	service number	source_addr
2	`destination_addr`	c-octet string	yes	subscribers MSISDN (phone number) or hashed value	destination_addr
3	`service_type`	c-octet string	no	message type	service_type
4	`data_coding`	integer	yes	type of content in field 'content' that is being sent to subscriber	data_coding
5	`sm_length`	integer	yes	defines field which contains content	sm_length
6	`short_message`	octet string	no	content that is being sent to subscriber	short_message
7	`message_payload`	tlv	no	content that is being sent to subscriber	message_payload

Table 17. submit_sm_resp PDU parameters detailed description

#	Name	Type	Required	Description	Ref
1	`message_id`	c-octet string	yes	message identifier	message_id
2	`command_status`	integer	yes	result of `submit_sm` PDU acceptance	command_status

4.2.2. Receive MO

SMPP PDU to receive MO message from CPA System

```
deliver_sm
```

SMPP PDU to inform CPA System on deliever_sm PDU acceptance

```
deliver_sm_resp
```

Table 18. deliver_sm PDU parameters detailed description

#	Name	Type	Required	Description	Ref
1	`receipted_message_id`	c-octet string	yes	subscriber's request identifier (UUID)	receipted_message_id
2	`service_type`	c-octet string	no	message type	service_type
3	`source_addr`	c-octet string	yes	subscribers MSISDN (phone number) or hashed value	source_addr
4	`destination_addr`	c-octet string	yes	service number	destination_addr
5	`esm_class`	integer	yes	defines request type: DLR or MO	esm_class
5	`data_coding`	integer	yes	type of content that is being sent to subscriber	data_coding
6	`sm_length`	integer	yes	defines field which contains content	sm_length
7	`message_payload`	tlv	no	content that is being sent to subscriber	message_payload

Table 19. deliver_sm_resp PDU parameters detailed description

#	Name	Type	Required	Description	Ref
1	`command_status`	integer	yes	result of `deliver_sm` PDU acceptance	command_status

4.2.3. Receive DLR

SMPP PDU to receive Delivery Report from CPA System

```
deliver_sm
```

SMPP PDU to inform CPA System on deliever_sm PDU acceptance

```
deliver_sm_resp
```

Table 20. deliver_sm PDU parameters detailed description

#	Name	Type	Required	Description	Ref
1	`receipted_message_id`	c-octet string	yes	message identifier (UUID)	receipted_message_id
2	`esm_class`	integer	yes	defines request type: DLR or MO	esm_class
3	`message_state`	tlv	yes	delivery report status	message_state

Table 21. deliver_sm_resp PDU parameters detailed description

#	Name	Type	Required	Description	Ref
1	`command_status`	integer	yes	result of `deliver_sm` PDU acceptance	command_status

Chapter 5. Parameters definition

5.1. REST API

5.1.1. Header parameters

Header is **mandatory** and has to contain 2 parameters:

- Authorization
- Content-Type

Authentication and authorization

Client Authentication is handled by the server using HTTP Basic Authentication where the client's username and password are concatenated in the form *<username>:<password>*, encoded using **base64**.

Consequently Content Provider should send its login and password in HTTP header as follows:

JSON/XML	`Authorization: Basic dm9yZGVsOnZvcmRlbA==`

Content-Type

The Content-Type entity is used to indicate the media type of the resource. In requests, the client tells the server what type of data is actually sent.

Consequently Content Provider should send its request data type in HTTP header as follows:

JSON	`Content-Type: application/json`
XML	`Content-Type: application/xml`

5.1.2. Mandatory parameters

date

Definition

Parameter definition depends on request type:

- for MO `date` defines time and date when CPA System received subscriber's request from External content delivery System;
- for DLR `date` defines time and date when CPA System received DLR from External content delivery System.

Features

1. Parameter is used for both `JSON` and `XML` data-interchange formats.

Format

Parameter format depends on data-interchange format.

`JSON`
: date format should be set as **TIMESTAMP (UTC) in milliseconds**.

  Example

Human time:	**Wednesday, January 31, 2018 12:45:15 PM**
TIMESTAMP (UTC) in ms:	**1517402715000**

`XML`
: date format should be set in format: **EEE, d MMM yyyy HH:mm:ss Z**, where:

  - `EEE` - abbreviated day of the week
  - `d` - day of the month
  - `MMM` - abbreviated month of the year
  - `yyyy` - year in four-digit format
  - `HH` - hour of the day (0-23)
  - `mm` - minute
  - `ss` - second
  - `Z` - UTC time offset; CPA system uses the term GMT instead of UTC in its definition of local time.

  Example

Human time:	**Wednesday, January 31, 2018 12:45:15 PM**
EEE, d MMM yyyy HH:mm:ss Z:	**Wed, 31 Jan 20018 12:45:15 GMT**

destination / sin

Definition

Parameter that defines message destination number; destination number where the message is sent.

Features

1. Parameter is used for both `JSON` and `XML` data-interchange formats.
2. Parameter name depends on data-interchange format.

`JSON`	`destination`
`XML`	`sin`

3. Parameter value depends on request type and data-interchange format.

   `JSON`
   - for MO `destination` defines Content Provider's service number;
   - for MT `destination` defines subscribers MSISDN (phone number).

   `XML`
   for both MO and MT `sin` defines defines subscribers MSISDN (phone number)

4. Parameter has no default value.

content / body

Definition

Parameter that defines content that is being sent; cannot be empty string.

Features

1. The *receiver* of content depends on request type:
   - for MO *content is being sent to Content Provider*;
   - for MT *content is being sent to subscriber*.
2. Parameter is used for both `JSON` and `XML` data-interchange formats.
3. Parameter name depends on data-interchange format.

`JSON`	`content`
`XML`	`body`

4. Parameter's length recommendations:
   a. for **sms / plain** message

`JSON`	`"bearerType": "sms"` `"contentType": "text/plain"`

- for `content` encoded to SMSCDefaultAlphabet, recommended length is <= 1000 characters;
- for `content` encoded to UCS2(ISO/IEC-10646), recommended length is <= 499 characters.

5. Parameter has no default value.

mid

Definition

Message identifier - a string used to identify a particular Mobile terminated content response. Parameters `mid` are used by CPA System and Content Provider in all messages concerning a particular Mobile terminated content response.

Format

UUID

Features

1. Parameter is used for both `JSON` and `XML` data-interchange formats.
2. Parameter origin depends on request's data-interchange format:

   `JSON`
   : if MT has `JSON` format, than `mid` is generated by CPA System and sent back to Content Provider in acceptance response.

   `XML`
   : if MT has `XML` format, than `mid` is generated by Content Provider and sent to CPA System within MT response.

3. Parameter has no default value.

nodeId

Definition

Parameter that defines CPA System's node identifier.

Features

1. Parameter is used only for `XML` data-interchange format.
2. Parameter is used only for request type DLR.
3. Parameter value should be always set to `0`.
4. Parameter has no default value.
5. Parameter is not processed by CPA System but should be set in request.

service

Definition

Parameter that defines request type: Delivery report or Mobile originated content request.

Features

1. Parameter is used only for `XML` data-interchange format.
2. Parameter value depends on request type:
   - for MO `service` always takes value `content-request`;
   - for DLR `service` always takes value `delivery-report`.
3. Parameter has no default value.

source / sn

Definition

Parameter that defines number of message origin; source number from where the message is received.

Features

1. Parameter is used for both `JSON` and `XML` data-interchange formats.
2. Parameter name depends on data-interchange format.

`JSON`	`source`
`XML`	`sn`

3. Parameter value depends on request type and data-interchange format.

   `JSON`
   - for MO `source` defines subscribers MSISDN (phone number);
   - for MT `source` defines Content Provider's service number.

   `XML`
   for both MO and MT `sn` defines Content Provider's service number.

4. Parameter has no default value.

status

Definition

Parameter that defines Delivery report status.

Features

1. Parameter is used for both `JSON` and `XML` data-interchange formats.
2. Parameter is used only for request type DLR.
3. Parameter value depends on data-interchange format.

   `JSON`
   `status` can take next values:
   - enroute

- delivered
- expired
- deleted
- undeliverable
- accepted
- unknown
- rejected

`XML`

`status` can take next values:

- Enroute
- Delivered
- Expired
- Deleted
- Undeliverable
- Accepted
- Unknown
- Rejected

4. Parameter has no default value.

report.status

Definition

Parameter that defines result of request acceptance for processing.

Features

1. Parameter is used only for `XML` data-interchange format.
2. Parameter value

   `report.status` can take next values:

   - Accepted - request accepted successfully
   - Rejected - request rejected

3. Parameter has no default value.

report.status.errorCode

Definition

Parameter that defines error identifier.

Features

1. Parameter is used only for `XML` data-interchange format.
2. Parameter values are provided in XML error types
3. Parameter has default value for success response.
   Default value is `0`.

5.1.3. Optional parameters

Restriction

Each request type has a set of mandatory and optional parameters. Must be taken into account the following: if optional parameter is included to request body, than it has to contain value, otherwise request is not being processed by CPA System.

bearerType / bearer

Definition

Parameter that defines message type.

Features

1. Parameter is used for both `JSON` and `XML` data-interchange formats.
2. Parameter name depends on data-interchange format.

`JSON`	`bearerType`
`XML`	`bearer`

3. Parameter value depends on data-interchange format.

   `JSON`
   `bearerType` can take next values:

   - sms
   - ussd

   `XML`
   `bearer` can take next values:

   - SMS
   - SMS0
   - USSD

4. Parameter has default value.
   Default value depends on data-interchange format.

`JSON`	`"bearerType": "sms"`
`XML`	`bearer="SMS"`

   If parameter is not set in request, then CPA System sets default value.

callbackNum

Definition

Parameter that defines Content Provider's callback number.
Parameter is used to set Content Provider's callback number, received in MT request, independently of Content Provider's initial configuration.

Features

1. Parameter is used only for `JSON` data-interchange format.
2. Parameter is allowed to be used only by preliminary agreement with Kyivstar JSC.
3. Parameter has no default value.

contentType / content-type

Definition

Parameter that defines type of content in parameter 'content' that is being sent.

Features

1. Parameter is used for both `JSON` and `XML` data-interchange formats.
2. Parameter name depends on data-interchange format.

`JSON`	`contentType`
`XML`	`content-type`

3. Parameter value depends on data-interchange format and message type.

   `JSON`

   `contentType` can take next values:

   - text/plain
   - text/zero
   - smpp/binary
   - text/ussn
   - text/ussr

     Definite value depends on message type (`bearerType`):

     a. for `"bearerType": "sms"`, `contentType` can be set to:
        - text/plain
        - text/zero
        - smpp/binary

     b. for `"bearerType": "ussd"`, `contentType` can be set to:
        - text/ussn

- text/ussr

`XML`

`content-type` can take next values:

- text/plain
- smpp/binary

  Definite value does not depend on message type (`bearerType`).

4. Parameter has default value.
   Default value depends on data-interchange format and message type.

   `JSON`

   Default value depends on message type:

for `"bearerType": "sms"`	**default value:** `"contentType": "text/plain"`
for `"bearerType": "ussd"`	**default value:** `"contentType": "text/ussn"`

   `XML`

   Default value does not depend on message type.
   Default value is `text/plain`.

   If parameter is not set in request, then CPA System sets default value.

eos

Definition

Parameter that defines end of session (interaction termination).

Features

1. Parameter is used only for `XML` data-interchange format.
2. Parameter is used only for USSD/MENU.
   Should be set only if `bearer="USSD"` in requested MT, sent as response to a particular MO.
3. Parameter value

   `eos` can take next values:

   - true - interaction should be terminated

4. Parameter has no default value.

errorId

Definition

Parameter that defines error identifier.

Features

1. Parameter is used only for `JSON` data-interchange format.
2. Parameter values are provided in JSON error types
3. Parameter has no default value.

errorMsg / error

Definition

Parameter that defines error message.

Features

1. Parameter is used for both `JSON` and `XML` data-interchange formats.
2. Parameter name depends on data-interchange format.

`JSON`	`errorMsg`
`XML`	`error`

3. Parameter value depends on data-interchange format.

   `JSON`
   : `errorMsg` values are provided in JSON error types.

   `XML`
   : `error` values are provided in XML error types.
   Can optionally contain error identifier.

4. Parameter has no default value.

report.status.error

Definition

Parameter that defines error message.

Features

1. Parameter is used only for `XML` data-interchange format.
2. Parameter values are provided in XML error types
3. Parameter has no default value.

rid

Definition

Request identifier - a string used to identify a particular Mobile originated content request.
Parameters `rid` are used by CPA System and Content Provider in all messages concerning a particular Mobile originated content request and requested Mobile terminated content response.

Format

UUID

Features

1. Parameter is used for both `JSON` and `XML` data-interchange formats.
2. Parameter `rid` is generated by CPA System and forwarded to Content Provider.
3. Parameter requiredness depends on request type:
    - for MO `rid` is **mandatory**;
    - for MT `rid` is **optional**. Content Provider should set `rid` in requested MT, sent as response to a particular MO.
4. Parameter has no default value.

serviceType / paid

Definition

Parameter that defines type of payment for the message.

Features

1. Parameter is used for both `JSON` and `XML` data-interchange formats.
2. Parameter is used only for request type MT.
3. Parameter name depends on data-interchange format.

`JSON`	`serviceType`
`XML`	`paid`

4. Parameter value does not depend on data-interchange format.

   `serviceType` / `paid` can take next values:

    - 'false' - message is free;
    - 'true', '1000' - message is paid by subscriber;
    - '2xxx' - message is paid by Content Provider

5. Parameter has default value.
   Default value is '**false**'.
   If parameter is not set in request, then CPA System sets default value.

5.2. SMPP Protocol

CPA System processes SMPP PDUs parameters as closely as possible to Short Message Peer to Peer Protocol Specification v3.4.

Current chapter provides parameters description of CPA System SMPP Protocol implementation as follows:

- parameters that are implemented accurately to SMPP Protocol Specification v3.4 are not described but have links to specific sections;
- parameters that have custom implementation are described .

5.2.1. Header parameters

Header processing is implemented according to SMPP Protocol Specification v3.4.
For detailing Header Parameters section 5.1 'Command Header Parameters' may be considered.

5.2.2. Mandatory parameters

command_status

Definition

Parameter that defines result of SMPP request. For detailing section *5.1.3* `command_status` *of SMPP Protocol Specification v3.4* may be considered.

data_coding

Definition

Parameter that defines type of content that is being sent.

Features

1. Parameter values are provided in Data encoding scheme.
2. Processing
   Processing logic of `data_coding` value:
   - if `data_coding` is set as SMSCDefaultAlphabet or Latin1(ISO-8859-1) or UCS2(ISO/IEC-10646), than content type is proceed as `text/plain`;
   - if else, content type is proceed as `smpp/binary`.

destination_addr

Definition

Parameter that defines message destination number; destination number where the message is sent.

Features

1. Parameter value depends on request type:
   - for MO `destination_addr` defines Content Provider's service number;
   - for MT `destination_addr` defines subscribers MSISDN (phone number).
2. Parameter has no default value.

esm_class

Definition

Parameter that defines request type: Delivery report or Mobile originated content request.

Features

1. Parameter value depends on request type:
   - for MO `esm_class` always takes value `xx0000xx`;
   - for DLR `esm_class` always takes value `xx0001xx`.
2. Parameter has no default value.

message_id

Definition

Message identifier - a string used to identify a particular Mobile terminated content response. CPA System generates `message_id` and sends it back to Content Provider in submit_sm_resp PDU.

message_state

Definition

Parameter that defines Delivery report status. For detailing section *5.2.28* `message_state` *of SMPP Protocol Specification v3.4* may be considered.

service_type

Definition

Parameter that defines message type.

Features

1. Parameter has defined set of values.
   `service_type` can take next values:
   - SMS
   - USSD
2. Parameter has default value.
   Default value is `SMS`.
   If parameter is not set in request, then CPA System sets default value.

sm_length

Definition

Parameter that defines field which contains content. For detailing section *5.2.21* `sm_length` *of SMPP Protocol Specification v3.4* may be considered.

source_addr

Definition

Parameter that defines number of message origin; source number from where the message is received.

Features

1. Parameter value depends on request type:
   - for MO `source_addr` defines subscribers MSISDN (phone number);
   - for MT `source_addr` defines Content Provider's service number.
2. Parameter has no default value.

5.2.3. Optional parameters

message_payload

Definition

Parameter that defines content that is being sent.
For detailing section *5.3.2.32* `message_payload` *of SMPP Protocol Specification v3.4* may be considered.

Features

1. The *receiver* of content depends on request type:
    - for MO content is being sent to Content Provider;
    - for MT content is being sent to subscriber.

receipted_message_id

Parameter definition and features depends on request type.

Mobile originated content request

Definition

For Mobile originated content request `receipted_message_id` defines *subscriber's request identifier*.

Features

1. Parameter is generated by CPA System and forwarded to Content Provider.
2. Parameter is **mandatory**.
3. Parameter has no default value.

Delivery report

Definition

For Delivery Report `receipted_message_id` defines *message identifier* of particular MT for which Delivery report is sent.

Features

1. Parameter is **mandatory**.
2. Parameter has no default value.

short_message

Definition

Parameter that defines content that is being sent.
For detailing section *5.2.22* `short_message` *of SMPP Protocol Specification v3.4* may be considered.

Features

1. The *receiver* of content depends on request type:
    - for MO content is being sent to Content Provider;
    - for MT content is being sent to subscriber.

Chapter 6. Message types

Current chapter provides syntax notation examples for REST API only

CPA System provides an opportunity to transfer different informational services from Content provider to subscriber and vice verse by using different message and content types. One message type can support several content types.

Message type is defined by parameter:

JSON	bearerType
XML	bearer

Content type is defined by parameter:

JSON	contentType
XML	content-type

Detailed description, syntax and examples of different message types with relevant content types are provided in next sections.

6.1. SMS message

SMS message (short message; SMS message) - a brief message sent to / from subscriber through the Short Message Service (SMS).

CPA System allows to use both `JSON` and `XML` data-interchange formats for SMS messages

Within interaction between CPA System and Content Provider **SMS message** can be defined by using following parameters:

JSON	"bearerType": "sms"
XML	bearer="SMS"

CPA System provides an opportunity to use following SMS message types:

1. plain
2. binary
3. silent

Detailed description is provided in next sections.

6.1.1. SMS / plain

SMS / **plain** (plain short message; plain SMS message) - readable short message, represented by standard ASCII characters, including numbers, symbols, and spaces.

Plain short message: `JSON`

SMS / **plain** can be defined by using following parameters:

```
JSON        "bearerType": "sms"
            "contentType": "text/plain"
```

However, it is not necessary to set following parameters:

- `"bearerType": "sms"` and `"contentType": "text/plain"`
  as they are set by default if `bearerType` and `contentType` are absent in request.

MT SMS / plain HTTP request body example

```
{
    "source": "101999",
    "destination": "380670000001",
    "content": "Hello World!"
}
```

Plain short message: `XML`

SMS / **plain** can be defined by using following parameters:

```
XML         bearer="SMS"
            content-type="text/plain"
```

However, it is not necessary to set following parameters:

- `bearer="SMS"` and `content-type="text/plain"`
  as they are set by default if `bearer` and `content-type` are absent in request.

MT SMS / plain HTTP request body example

```
<message mid="904705f0-3c5e-4556-8339-81b7fd2e3d83">
    <sn>101999</sn>
    <sin>380670000001</sin>
    <body>Hello World!</body>
</message>
```

Data parameters detailed description

MT HTTP request body data parameters detailed descriptions are provided in sections:

- JSON: Send MT;
- XML: Send MT.

6.1.2. SMS / silent

SMS / silent (silent short message; silent SMS message; short message type 0; SMS0) - a special type of short message that is indicated neither on the display nor by an acoustic signal. SMS / silent is not saved in phones memory and does not contain any content. Is frequently used to allow SMSC perform subscriber's core balance charging.

Silent short message: `JSON`

SMS / silent can be defined by using following parameters:

```
JSON        "bearerType": "sms"
            "contentType": "text/zero"
```

However, it is not necessary to set following parameter:

- `"bearerType": "sms"` - as it is set by default if `bearerType` is absent in request.

MT SMS / silent HTTP request body example

```
{
    "source": "101999",
    "destination": "380670000001",
    "contentType": "text/zero"
}
```

Silent short message: `XML`

SMS / silent can be defined by using following parameters:

```
XML         bearer="SMS0"
            content-type="text/plain"
```

MT SMS / silent HTTP request body example

```
<message mid="904705f0-3c5e-4556-8339-81b7fd2e3d83" paid="false" bearer="SMS0">
    <sn>101999</sn>
    <sin>380670000001</sin>
    <body content-type="text/plain"></body>
</message>
```

Data parameters detailed description

MT HTTP request body data parameters detailed descriptions are provided in sections:

- JSON: Send MT;
- XML: Send MT.

6.1.3. SMS / binary

SMS / **binary** (binary short message; binary SMS message) - a binary short message; commonly not viewable by phone as text messages and used for data (e.g. images and ringing tones) or installation messages.

Binary short message: `JSON`

SMS / **binary** can be defined by using following parameters:

```
JSON        "bearerType": "sms"
            "contentType": "smpp/binary"
```

However, it is not necessary to set following parameter:

- `"bearerType": "sms"` - as it is set by default if `bearerType` is absent in request.

MT SMS / binary HTTP request body example

```
{
    "source": "101999",
    "destination": "380670000001",
    "contentType": "smpp/binary"
    "content": "SSBrbm93IHdoYXQgeW91IGdvb2dsZSBsYXN0IG5pZ2h0IEhBSEE="
}
```

Binary short message: `XML`

SMS / **binary** can be defined by using following parameters:

```
XML         bearer="SMS"
            content-type="smpp/binary"
```

MT SMS / binary HTTP request body example

```
<message mid="904705f0-3c5e-4556-8339-81b7fd2e3d83" paid="false" bearer="SMS">
    <sn>101999</sn>
    <sin>380670000001</sin>
    <body content-type="smpp/binary">SSBzYXcgeW91ciB0b3lz=</body>
</message>
```

Data parameters detailed description

MT HTTP request body data parameters detailed descriptions are provided in sections:

- JSON: Send MT;
- XML: Send MT.

6.2. USSD message

CPA System provides an opportunity to use following USSD operations:

1. USSD / push
2. USSD / menu

Detailed description is provided in next sections.

6.2.1. USSD / push

USSD / push operation intends monodirectional message sending. Content Provider can send message which requires no subscriber's response. In current document is used term **'Unstructured Supplementary Services Notification' (USSD notification, USSN message, USSN)** to define USSD / push message with informational service which requires no response.

CPA System supports only `JSON` data-interchange format for USSD / push

USSN message: `JSON`

Within interaction between CPA System and Content Provider **USSN message** can be defined by using following parameters:

```
JSON        "bearerType": "ussd"
            "contentType": "text/ussn"
```

However, it is not necessary to set following parameter:

- `"contentType": "text/ussn"` - as it is set by default if `contentType` is absent in request.

Consequently Content Provider can send USSN message as provided in following example.

MT USSD / push HTTP request body example

```
{
    "source": "101999",
    "destination": "380670000001",
    "bearerType": "ussd",
    "content": "World enddate is tomorrow!"
}
```

Data parameters detailed description

MT HTTP request body data parameters detailed description is provided in section:

- JSON: Send MT.

6.2.2. USSD / menu

USSD / menu operation intends bidirectional interactive interaction between subscriber and Content Provider in the mode of sending messages. Subscriber can initiate USSD / menu operation by sending USSD message, which requires response, to Content Provider in order to receive informational service. Content Provider responses to such message. If necessary, several message exchange can be performed. In current document is used term **'Unstructured Supplementary Services Request' (USSD request, USSR message, USSR)** to define message which requires response. Within USSD / menu interaction, final message indicates interaction termination and requires no response. Other words, final message is expected to be a USSN message.

CPA System allows to use both `JSON` and `XML` data-interchange formats for USSD / menu

USSD message

USSD message (from subscriber's viewpoint) - a code that consist of digits and special signs. A typical USSD message starts with an asterisk `*`, followed by digits that comprise commands or data. Groups of digits may be separated by additional asterisks. The message is terminated with a sign `#`.

USSD message format: `*XXX*YYY#`, where:
`XXX` – USSD code
`YYY` – USSD command; may contain digits and asterisks

Content Provider receives USSD command (`YYY`) in parameter `content`.

USSD / menu interaction flow

Generally USSD / menu interaction is initiated by subscriber in order to receive informational service. Although it is also possible that USSD / menu interaction may be initiated by Content Provider.

USSD / menu interaction initiated by subscriber is performed as follows:

1. Subscriber sends USSR message.
2. Content Provider responses. Within responding 2 scenarios can be performed:
    a. Content Provider's message requires subscriber's response (USSR message) in order to continue interaction. (If necessary, several message exchange can be performed.)
    b. Content Provider's message is final, terminates interaction, and requires no subscriber's response (USSN message).

For deeper detailing of USSD / menu initiated by subscriber for both `JSON` and `XML` data-interchange formats next sections may be considered.

USSD / menu: JSON

USSR message can be defined by using following parameters:

```
JSON        "bearerType": "ussd"
            "contentType": "text/ussr"
```

1. **Subscriber sends USSR message**

MO USSR HTTP request body example

```
{
    "rid": "d8549def-5d93-4e52-8b63-b282ccefa971",
    "date": "1513296000000",
    "source": "380670000001",
    "destination": "101999",
    "bearerType": "ussd",
    "contentType": "text/ussr",
    "content": "666"
}
```

2. **Content Provider responses**

2.a. Content Provider's **message requires subscriber's response** in order to continue interaction.

MT USSR HTTP request body example

```
{
    "rid": "d8549def-5d93-4e52-8b63-b282ccefa971",
    "source": "101999",
    "destination": "380670000001",
    "bearerType": "ussd",
    "contentType": "text/ussr",
    "content": "To know World enddate enter *13666#"
}
```

2.b. Content Provider's **message is final**, terminates interaction, and requires no subscriber's response.

Final message requires no response and indicates interaction termination. Other words, final message is expected to be a USSN message. Subscriber cannot answer on USSN message. Detailed description of USSN message is provided in section (USSD / push).

Data parameters detailed description

MT/MO HTTP request body data parameters detailed descriptions are provided in sections:

- JSON: Send MT;
- JSON: Receive MO.

USSD / menu: `XML`

USSR message can be defined by using following parameters:

`XML`	`bearer="USSD"` `content-type="text/plain"`

1. **Subscriber sends USSR message**

MO USSR HTTP request body example

```
<message rid="d8549def-5d93-4e52-8b63-b282ccefa971" bearer="USSD">
    <sn>101999</sn>
    <sin>380670000001</sin>
    <service>content-request</service>
    <status date="Wed, 31 Jan 20018 16:20:00 GMT"/>
    <body content-type="text/plain">666</body>
</message>
```

2. **Content Provider responses**

2.a. Content Provider's **message requires subscriber's response** in order to continue interaction.

MT USSR HTTP request body example

```
<message rid="d8549def-5d93-4e52-8b63-b282ccefa971"
    mid="904705f0-3c5e-4556-8339-81b7fd2e3d83" paid="false" bearer="USSD">
    <sn>101999</sn>
    <sin>380670000001</sin>
    <body content-type="text/plain">To know World enddate enter *13666#</body>
</message>
```

2.b. Content Provider's **message is final**, terminates interaction, and requires no subscriber's response. MT USSN message should contain parameter that indicates interaction termination - `eos`.

MT USSN HTTP request body example

```
<message rid="d8549def-5d93-4e52-8b63-b282ccefa971"
    mid="904705f0-3c5e-4556-8339-81b7fd2e3d83" paid="false" bearer="USSD" eos="true">
    <sn>101999</sn>
    <sin>380670000001</sin>
    <body content-type="text/plain">World enddate is tomorrow!</body>
</message>
```

Data parameters detailed description

MT/MO HTTP request body data parameters detailed descriptions are provided in sections:

- XML: Send MT;
- XML: Receive MO.

6.3. Message encoding

CPA System processes different character encoding types for content that is being transferred from subscriber to Content Provider and vice verse. Encoding type depends on message content type, and is detailed in following sections. Definite parameters' values that define data encoding scheme are provided in section Data encoding scheme in accordance with:

- Short Message Peer to Peer Protocol Specification v3.4;
- 3GPP TS 23.038 version 15.0.0.

6.3.1. Encoding for text/plain content type

Encoding for DLR, MO messages with text/plain content type

CPA System accepts from External content delivery system DLR and MO messages encoded to:

- SMSCDefaultAlphabet;
- UCS2(ISO/IEC-10646).

CPA System sends to Content Provider DLR and MO messages encoded to:

- UTF-8.

Encoding for MT message with text/plain content type

CPA System accepts from Content Provider MT message and encodes it, depending on characters in content, to:

- SMSCDefaultAlphabet - if content has latin characters only;
- UCS2(ISO/IEC-10646) - if content has at least one cyrillic character.

6.3.2. Encoding for smpp/binary content type

Encoding for MO message with smpp/binary content type

CPA System accepts from External content delivery system MO message encoded to:

- Octet unspecified (8-bit binary);
- Octet unspecified (8-bit binary) ALL TECHNOLOGIES;
- Class 0 (Flash message);
- Class 1 (ME-specific);
- Class 2 (SIM/USIM-specific);
- Class 3 (TE-specific);
- Class 0 (Flash message); bit 3 is reserved;
- Class 1 (ME-specific); bit 3 is reserved;

- Class 2 (SIM/USIM-specific); bit 3 is reserved;
- Class 3 (TE-specific); bit 3 is reserved.

CPA System forwards to Content Provider MO message encoded using base64.

Encoding for MT message with smpp/binary content type

CPA System accepts from Content Provider MT message encoded using base64.

Chapter 7. General

7.1. HTTP verbs

Content Provider Access System tries to adhere as closely as possible to standard HTTP and REST conventions in its use of HTTP verbs.

Verb	Usage
`GET`	Used to retrieve data from a specified resource. The data requested from the server with GET can be in `JSON` format
`POST`	POST is used to send data to a server to create a resource. The data sent to the server with POST can be in `JSON` or `XML` formats

7.2. HTTP status codes

Content Provider Access System tries to adhere as closely as possible to standard HTTP and REST conventions in its use of HTTP status codes.

Status code	**Usage**
`200 OK`	Standard response for successful HTTP requests. The actual response will depend on the request method used. In a GET request, the response will contain an entity corresponding to the requested resource. In a POST request, the response will contain an entity describing or containing the result of the action.
`202 Accepted`	The request has been accepted for processing, but the processing has not been completed. The request might or might not be eventually acted upon, and may be disallowed when processing occurs.
`400 Bad Request`	The server cannot or will not process the request due to something that is perceived to be a client error (e.g., malformed request syntax, invalid request message framing, or deceptive request routing).
`401 Unauthorized`	Is used when authentication is required and has failed or has not yet been provided.
`403 Forbidden`	The request was valid, but the server is refusing action. The user might not have the necessary permissions for a resource, or may need an account of some sort.
`404 Not Found`	The requested resource could not be found but may be available again in the future. Subsequent requests by the client are permissible.
`500 Internal Error`	The server encountered an unexpected condition which prevented it from fulfilling the request.
`502 Bad Gateway`	The server was acting as a gateway or proxy and received an invalid response from the upstream server.

7.3. Error information

Description of all errors. Can be developed and supplemented. Error identifier and message depends on interface and message format.

JSON error types

ID	Text	Description
1007	Request not found	Request not found
1014	Invalid protocol	Invalid protocol
1016	Unknown bearer type	Content Provider's message contains unknown bearer type
1018	Invalid source address	Content Provider's message contains invalid source address
1019	Invalid destination address	Content Provider's message contains invalid destination address (subscriber's phone number)
1040	Resource not found	Provided resource is invalid
1041	Destination has wrong length 12… 21(msisdn) or 32(hash)	Provided destination has wrong length 12… 21(msisdn) or 32(hash)
1042	Destination must contain only digits	Provided destination contains illegal characters
1043	Content is absent	Content Provider's message contains no content
1045	Messages not allowed from source: XXX	Defined source is restricted to send messages
1051	Content have illegal characters: XXX	Provided content contains illegal characters
1053	The subscriber does not have enough money	Subscriber's account is low than tariff for message that requires charging
1054	It is not Kyivstar subscriber	Subscriber is not a customer of Kyivstar
1056	Message is not allowed with provided parameters	Content Provider's message contains parameter that is not allowed with provided value
1057	Message expired	Message defined by `rid` has already expired
1058	Unknown content type	Content Provider's message contains unknown content type
1059	All allowed free messages sent	Request defined by `rid` has run out number of free messages
1060	All allowed paid messages sent	Request defined by `rid` has run out number of paid messages
1062	Provided prefix is not allowed: %PARAM%	Provided country and national destination codes are not allowed according to E.164

ID	Text	Description
1063	Hashed destination is not allowed	Provided hashed destination is not allowed
1064	Msisdn destination is not allowed	Provided unhashed destination is not allowed
1100	Internal CPA error	Internal CPA System error
1200	Internal CPA error	Internal CPA System error
1300	Internal CPA error	Internal CPA System error
1400	Internal CPA error	Internal CPA System error
1500	Internal CPA error	Internal CPA System error
1901	Internal CPA error	Internal CPA System error

XML error types

ID	Text	Description
1	No request messages are not allowed	Content Provider is not allowed to send messages without request in this source-destination-bearer combination
2	Invalid RID: XXX	CPA System was unable to parse message RID
3	Message not found	CPA System was unable to find requested delivery report. Delivery state is unknown
4	Service not found: XXX	CPA System was unable to find service description which delivery report should be sent to. This can happen when service was deleted
5	Content type unknown	CPA System was unable to define content type (SMS or USSD)
6	Source/destination not allowed for this provider	CPA System was unable to find service corresponding to source-destination-bearer in case of content message without request
7	Request not found	CPA System was unable to find request defined by RID. This can happens when request was deleted from CPA System database
8	Message expired	Request defined by RID has already expired
9	Incorrect charge-id: XXX	CPA System was unable to parse charge ID
10	All allowed free messages sent	Request defined by RID has run out number of free messages
11	All allowed charged messages sent	Request defined by RID has run out number of charged messages

ID	Text	Description
12	Charge-id not allowed: XXX	CPA System was unable to find tariff code appropriate to requested charge ID set by Content Provider in attribute paid for this service
13	Low balance	Subscriber’s account is low than tariff for message that requires charging
14	Invalid protocol	Protocol’s error
15	Throttle exceeded	Content Provider exceeded allowed speed of sending messages to CPA System
16	Bearer unknown	Content Provider defined unknown bearer in message
17	Parse XML error	Content Provider provided XML document that cannot be parsed
18	Invalid source address	Provided source address is invalid
19	Invalid destination address	Provided destination address (subscriber’s phone number) is invalid
20	Request not delivered to provider	Content Provider tried to send message with RID that was not accepted by Content Provider. Usually this happens when Content Provider received request but failed to answer 'Accepted'
21	Charge-id XXX not allowed for provider	CPA was unable to find tariff code appropriate to requested charge ID set by Content Provider in attribute 'paid' for this provider
22	Response not found	Message for requested DLR is not found (may be not accepted by CPA System)
23	Subscriber error	Subscriber type cannot be resolved
24	Subscriber billing error	Exception occurred in process of reservation money for charging on postpaid subscriber’s account
25	Delivery request was locked by CPA-support.	You need to contact with your account manager to resolve this error
26	Amount of non-request messages per day exceeded	You need to contact with your account manager to resolve this error
27	Amount of non-request messages per month exceeded	You need to contact with your account manager to resolve this error
28	Wrong subscriber type	This type of subscriber cannot receive the content (for example corporate subscriber cannot receive funbroker melodies)

ID	Text	Description
29	Melody already exists	Funbroker melody cannot be delivered to subscriber, because melody already exists in subscriber's profile
30	Content not found	Provided content ID does not exist is Content Provider's account
31	Content not active	Provided content ID is not in active state
32	Forbidden by user	Content/service is forbidden by user
33	Application in restricted mode	During making backup some functions of CPA System may be blocked
34	Other unexpected new error	Internal CPA System error
35	Content-type forbidden	Content-type specification is not allowed
36	Message locked	RID is temporary locked
100	Internal CPA error	Internal CPA System error
101	Internal transit error	Internal transit error

SMPP error types

Error code	Value	Description
ESME_RSYSERR	0x00000008	System Error
ESME_RINVSRCADR	0x0000000A	Invalid Source Address
ESME_RINVDSTADR	0x0000000B	Invalid Dest Addr
ESME_RINVNUMMSGS	0x00000055	Invalid number of messages
ESME_RINVEXPIRY	0x00000062	Invalid message validity period (Expiry time)
ESME_RINVPARLEN	0x000000C2	Invalid Parameter Length
ESME_RINVSERTYP	0x00000015	Invalid Service Type
ESME_RSYSERR	0x00000008	System Error

7.4. Data encoding scheme

Current section provides definite parameters values of data encoding scheme. Encoding for different message content types is provided in section Message encoding.

Bits [76543210]	Hexadecimal value	Meaning
text/plain		
00000000	0x00	SMSCDefaultAlphabet
00001000	0x08	UCS2(ISO/IEC-10646)
smpp/binary		
00000010	0x02	Octet unspecified (8-bit binary)
00000100	0x04	Octet unspecified (8-bit binary) ALL TECHNOLOGIES
11110100	0xf4	Class 0 (Flash message)
11110101	0xf5	Class 1 (ME-specific)
11110110	0xf6	Class 2 (SIM/USIM-specific)
11110111	0xf7	Class 3 (TE-specific)
11111100	0xfc	Class 0 (Flash message); bit 3 is reserved
11111101	0xfd	Class 1 (ME-specific); bit 3 is reserved
11111110	0xfe	Class 2 (SIM/USIM-specific); bit 3 is reserved
11111111	0xff	Class 3 (TE-specific); bit 3 is reserved

7.5. Glossary

A glossary defines key terms and abbreviations relevant to Content Provider Access System for the purpose of current document.

Content Provider Access System (CPA System; CPA)

a system that enables interaction between subscriber and Content Provider for informational services transferring.

Content Provider (Provider, CP)

a system that provides informational services to subscriber.

Delivery Report (DLR)

message to Content Provider to inform whether content was delivered to subscriber.

External Content Delivery System

an external system that directly sends/receives informational services to/from subscriber (e.g., SMSC)

External Short Message Entity

an external system that connects to a Short Message Service Center (SMSC) to engage in the sending and/or receiving of Short Messages.

Informational Services (content)

a specific sort of information provided by Content Provider.

Short Message (SMS message; SM)

a brief text message sent to/from subscriber through the Short Message Service (SMS).

Short Message Service (SMS)

a text messaging service that uses standardized communication protocols to enable mobile devices to exchange short text messages via a Short Message Service Centre.

Short Message Service Centre (SMS Centre; SMSC)

a system that provides relaying and store-and-forwarding of a short message.

Unstructured Supplementary Service Data (USSD)

a communication protocol used by GSM cellular telephones to communicate with the mobile network operator’s computers.

Unstructured Supplementary Service Data Message (USSD message)

a code that consist of digits and special signs sent to/from subscriber through the Unstructured Supplementary Service Data protocol.

Message Identifier

a string used to identify a particular Mobile terminated content response.
Message identifiers are used by CPA System and Content Provider in all messages concerning a particular Mobile terminated content response.

REST	mid
SMPP	message_id

Mobile Originated Content Request (MO request; MO message; MO; content request)

message from subscriber to order informational services form Content Provider.

Mobile Terminated Content Response (MT response; MT message; MT; content response)

message from Content Provider in order to provide informational services to subscriber.

Request Identifier (subscriber's request identifier)

a string used to identify a particular Mobile originated content request.
Request identifiers are used by CPA System and Content Provider in all messages concerning a particular Mobile originated content request and requested Mobile terminated content response.

REST	rid
SMPP	receipted_message_id

Requested Mobile Terminated Content Response (requested MT response; requested MT message; requested MT)

message from Content Provider in order to provide informational services to subscriber that were preliminary ordered.