1. Classification.asn

```
Classification DEFINITIONS IMPLICIT TAGS ::=
BEGIN
EXPORTS TAGGED,
        sorm-message-session,
        sorm-message-trap,
        sorm-message-task,
        sorm-message-report,
        sorm-message-management,
        sorm-message-unformatted,
        sorm-message-filter,

        sorm-request-identifier-pager,
        sorm-request-identifier-pstn,
        sorm-request-identifier-gsm,
        sorm-request-identifier-cdma,
        sorm-request-identifier-data-network,
        sorm-request-identifier-voip,

        sorm-report-identifier-pager,
```

sorm-report-identifier-pstn,
sorm-report-identifier-gsm,
sorm-report-identifier-cdma,
sorm-report-identifier-data-network,
sorm-report-identifier-voip,

sorm-request-payment-bank-transaction,
sorm-request-payment-express-pays,
sorm-request-payment-terminal-pays,
sorm-request-payment-service-center,
sorm-request-payment-cross-account,
sorm-request-payment-telephone-card,
sorm-request-payment-balance-fillups,
sorm-request-payment-bank-division-transfer,
sorm-request-payment-bank-card-transfer,
sorm-request-payment-bank-account-transfer,

sorm-report-payment-bank-transaction,
sorm-report-payment-express-pays,
sorm-report-payment-terminal-pays,
sorm-report-payment-service-center,
sorm-report-payment-cross-account,
sorm-report-payment-telephone-card,
sorm-report-payment-balance-fillups,
sorm-report-payment-bank-division-transfer,
sorm-report-payment-bank-card-transfer,
sorm-report-payment-bank-account-transfer,

sorm-request-connection-pager,
sorm-request-connection-pstn,
sorm-request-connection-mobile,
sorm-request-connection-aaa-login,
sorm-request-connection-resource,
sorm-request-connection-email,
sorm-request-connection-im,
sorm-request-connection-voip,
sorm-request-connection-file-transfer,
sorm-request-connection-term-access,
sorm-request-connection-raw-flows,
sorm-request-connection-entrance,
sorm-request-connection-address-translations,
sorm-request-connection-sms,

sorm-report-connection-pager,
sorm-report-connection-pstn,
sorm-report-connection-mobile,

sorm-report-connection-ipdr-header,
sorm-report-connection-aaa-login,
sorm-report-connection-resource,
sorm-report-connection-email,
sorm-report-connection-im,
sorm-report-connection-voip,
sorm-report-connection-file-transfer,
sorm-report-connection-term-access,
sorm-report-connection-raw-flows,
sorm-report-connection-address-translations,
sorm-report-connection-entrance,
sorm-report-connection-sms,

sorm-request-dictionaries,
sorm-report-dictionary-bunches,
sorm-report-dictionary-basic-stations,
sorm-report-dictionary-roaming-partners,
sorm-report-dictionary-switches,
sorm-report-dictionary-gates,
sorm-report-dictionary-call-types,
sorm-report-dictionary-supplement-services,
sorm-report-dictionary-pay-types,
sorm-report-dictionary-termination-causes,
sorm-report-dictionary-ip-numbering-plan,
sorm-report-dictionary-phone-numbering-plan,
sorm-report-dictionary-doc-types,
sorm-report-dictionary-telcos,
sorm-report-dictionary-ip-data-points,
sorm-report-dictionary-special-numbers,
sorm-report-dictionary-bunches-map,
sorm-report-dictionary-mobile-subscriber-identity-plan,
sorm-report-dictionary-signal-point-codes,
sorm-request-presense,
sorm-report-presense-abonents,
sorm-report-presense-connections,
sorm-report-presense-payments,
sorm-report-presense-dictionaries,
sorm-report-presense-locations,

sorm-request-abonent-person,
sorm-request-abonent-organization,

sorm-report-abonent-abonent,
sorm-report-abonent-service,
sorm-report-abonent-person,
sorm-report-abonent-organization,

```
        sorm-request-location,
        sorm-report-data-content-raw;

TAGGED ::= CLASS {
&id ObjectDescriptor UNIQUE,
&Data
}
WITH SYNTAX {
   OID &id
   DATA &Data
}

--- Классификация

OID ::= ObjectDescriptor

-- Подструктура сообщений
sorm-message-session OID ::= "280"
sorm-message-trap OID ::= "281"
sorm-message-task OID ::= "282"
sorm-message-report OID ::= "283"
sorm-message-management OID ::= "284"
sorm-message-unformatted OID ::= "285"
sorm-message-filter OID ::= "286"

-- Идентификаторы
sorm-request-identifier-pager OID ::= "140"
sorm-request-identifier-pstn OID ::= "141"
sorm-request-identifier-gsm OID ::= "142"
sorm-request-identifier-cdma OID ::= "143"
sorm-request-identifier-data-network OID ::= "144"
sorm-request-identifier-voip OID ::= "145"

sorm-report-identifier-pager OID ::= "1"
sorm-report-identifier-pstn OID ::= "2"
sorm-report-identifier-gsm OID ::= "3"
sorm-report-identifier-cdma OID ::= "4"
sorm-report-identifier-data-network OID ::= "5"
sorm-report-identifier-voip OID ::= "6"

-- Параметры соединений
sorm-request-connection-pager OID ::= "160"
sorm-request-connection-pstn OID ::= "161"
sorm-request-connection-mobile OID ::= "162"
sorm-request-connection-aaa-login OID ::= "164"
sorm-request-connection-resource OID ::= "165"
```

```
sorm-request-connection-email OID ::= "166"
sorm-request-connection-im OID ::= "167"
sorm-request-connection-voip OID ::= "168"
sorm-request-connection-file-transfer OID ::= "169"
sorm-request-connection-term-access OID ::= "170"
sorm-request-connection-raw-flows OID ::= "171"
sorm-request-connection-entrance OID ::= "172"
sorm-request-connection-address-translations OID ::= "173"
sorm-request-connection-sms OID ::= "174"

sorm-report-connection-pager OID ::= "20"
sorm-report-connection-pstn OID ::= "21"
sorm-report-connection-mobile OID ::= "22"
sorm-report-connection-ipdr-header OID ::= "23"
sorm-report-connection-aaa-login OID ::= "24"
sorm-report-connection-resource OID ::= "25"
sorm-report-connection-email OID ::= "26"
sorm-report-connection-im OID ::= "27"
sorm-report-connection-voip OID ::= "28"
sorm-report-connection-file-transfer OID ::= "29"
sorm-report-connection-term-access OID ::= "30"
sorm-report-connection-raw-flows OID ::= "31"
sorm-report-connection-address-translations OID ::= "32"
sorm-report-connection-entrance OID ::= "33"
sorm-report-connection-sms OID ::= "34"

-- Абоненты
sorm-request-abonent-person OID ::= "180"
sorm-request-abonent-organization OID ::= "181"

sorm-report-abonent-abonent OID ::= "40"
sorm-report-abonent-service OID ::= "41"
sorm-report-abonent-person OID ::= "42"
sorm-report-abonent-organization OID ::= "43"

-- Местоположение
sorm-request-location OID ::= "200"

-- Платежи
sorm-request-payment-bank-transaction OID ::= "220"
sorm-request-payment-express-pays OID ::= "221"
sorm-request-payment-terminal-pays OID ::= "222"
sorm-request-payment-service-center OID ::= "223"
sorm-request-payment-cross-account OID ::= "224"
sorm-request-payment-telephone-card OID ::= "225"
sorm-request-payment-balance-fillups OID ::= "226"
```

sorm-request-payment-bank-division-transfer OID ::= "227"
sorm-request-payment-bank-card-transfer OID ::= "228"
sorm-request-payment-bank-account-transfer OID ::= "229"

sorm-report-payment-bank-transaction OID ::= "80"
sorm-report-payment-express-pays OID ::= "81"
sorm-report-payment-terminal-pays OID ::= "82"
sorm-report-payment-service-center OID ::= "83"
sorm-report-payment-cross-account OID ::= "84"
sorm-report-payment-telephone-card OID ::= "85"
sorm-report-payment-balance-fillups OID ::= "86"
sorm-report-payment-bank-division-transfer OID ::= "87"
sorm-report-payment-bank-card-transfer OID ::= "88"
sorm-report-payment-bank-account-transfer OID ::= "89"

-- Справочники
sorm-request-dictionaries OID ::= "240"

sorm-report-dictionary-bunches OID ::= "100"
sorm-report-dictionary-basic-stations OID ::= "101"
sorm-report-dictionary-roaming-partners OID ::= "102"
sorm-report-dictionary-switches OID ::= "103"
sorm-report-dictionary-gates OID ::= "104"
sorm-report-dictionary-call-types OID ::= "105"
sorm-report-dictionary-supplement-services OID ::= "106"
sorm-report-dictionary-pay-types OID ::= "107"
sorm-report-dictionary-termination-causes OID ::= "108"
sorm-report-dictionary-ip-numbering-plan OID ::= "109"
sorm-report-dictionary-phone-numbering-plan OID ::= "110"
sorm-report-dictionary-doc-types OID ::= "111"
sorm-report-dictionary-telcos OID ::= "112"
sorm-report-dictionary-ip-data-points OID ::= "113"
sorm-report-dictionary-special-numbers OID ::= "114"
sorm-report-dictionary-bunches-map OID ::= "115"
sorm-report-dictionary-mobile-subscriber-identity-plan OID ::= "116"
sorm-report-dictionary-signal-point-codes OID ::= "132"

-- Запрос о наличии данных
sorm-request-presense OID ::= "260"

sorm-report-presense-abonents OID ::= "120"
sorm-report-presense-connections OID ::= "121"
sorm-report-presense-payments OID ::= "122"
sorm-report-presense-dictionaries OID ::= "123"
sorm-report-presense-locations OID ::= "124"

```
-- Запрос о содержимом потоков
sorm-report-data-content-raw OID ::= "50"

END
```

2. Addresses.asn

```
Addresses DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS AddressType,
        ReportedAddresses,
        ReportedAddress,
        RequestedAddress;

AddressType ::=   ENUMERATED {
registered (0),          --- для физических лиц – адрес регистрации по месту жительства (пребывания), для юридических лиц – адрес юридического лица в пределах места нахождения
postal (1),              --- почтовый адрес (дополнительный адрес для юридических лиц)
invoice (2),             --- адрес доставки счета (дополнительный адрес для юридических лиц)
device-location (3),     --- адрес установки пользовательского устройства (телефонного аппарата)
reserved (4)             --- резерв
}

ReportedAddresses ::= SEQUENCE OF ReportedAddress

ReportedAddress ::= SEQUENCE {
title   AddressType,                    --- тип адреса
address-info AddressInfoReport          --- адрес
}

AddressInfoReport ::= CHOICE {
struct-info[1]      AddressStructInfoReport,      --- структурированный адрес
unstruct-info[2]    UTF8String(SIZE (1 .. 1024)) --- неструктурированный адрес
}

AddressStructInfoReport ::= SEQUENCE {
 zip [0]            UTF8String (SIZE (1 .. 32)) OPTIONAL,  --- почтовый индекс, zip-код
 country [1] UTF8String (SIZE (1 .. 128)) OPTIONAL,        --- страна
 region [2]  UTF8String (SIZE (1 .. 128)) OPTIONAL,        --- область
```

```
        zone [3]          UTF8String (SIZE (1 .. 128)) OPTIONAL, ---        район,
муниципальный округ
        city [4]          UTF8String (SIZE (1 .. 128)) OPTIONAL, ---        город,
поселок, деревня
        street [5]   UTF8String (SIZE (1 .. 128)) OPTIONAL,        --- улица
        building [6]      UTF8String (SIZE (1 .. 128)) OPTIONAL, ---          дом,
строение
        build-sect [7]    UTF8String (SIZE (1 .. 128)) OPTIONAL, --- корпус
        apartment [8]     UTF8String (SIZE (1 .. 128)) OPTIONAL  ---    квартира,
офис
    }


    -- поля адресных данных
    RequestedAddress ::= SEQUENCE {
    zip [0]           UTF8String (SIZE (1 .. 32)) OPTIONAL,   ---    почтовый
индекс, zip-код
    country [1]   UTF8String (SIZE (1 .. 128)) OPTIONAL,        --- страна
    region [2]    UTF8String (SIZE (1 .. 128)) OPTIONAL,        --- область
    zone [3]          UTF8String (SIZE (1 .. 128)) OPTIONAL, ---        район,
муниципальный округ
    city [4]          UTF8String (SIZE (1 .. 128)) OPTIONAL, ---        город,
поселок, деревня, населенный пункт
    street [5]    UTF8String (SIZE (1 .. 128)) OPTIONAL,        --- улица
    building [6] UTF8String (SIZE (1 .. 128)) OPTIONAL,         ---          дом,
строение
    build-sect [7]    UTF8String (SIZE (1 .. 128)) OPTIONAL, --- корпус
    apartment [8]     UTF8String (SIZE (1 .. 128)) OPTIONAL  ---    квартира,
офис
    }

    END
```

3. Dictionaries.asn

```
    Dictionaries DEFINITIONS IMPLICIT TAGS ::=
    BEGIN

    EXPORTS
      TelcoID,
      TelcoList,

      DictionaryTask,
      DictionaryReport,

      PhoneAbonentType;
```

```
IMPORTS DateAndTime
  FROM Sorm
    TAGGED,
    sorm-request-dictionaries,
    sorm-report-dictionary-telcos,
    sorm-report-dictionary-bunches,
    sorm-report-dictionary-basic-stations,
    sorm-report-dictionary-roaming-partners,
    sorm-report-dictionary-switches,
    sorm-report-dictionary-gates,
    sorm-report-dictionary-call-types,
    sorm-report-dictionary-supplement-services,
    sorm-report-dictionary-pay-types,
    sorm-report-dictionary-termination-causes,
    sorm-report-dictionary-ip-numbering-plan,
    sorm-report-dictionary-phone-numbering-plan,
    sorm-report-dictionary-doc-types,
    sorm-report-dictionary-ip-data-points,
    sorm-report-dictionary-special-numbers,
    sorm-report-dictionary-bunches-map,
    sorm-report-dictionary-mobile-subscriber-identity-plan,
    sorm-report-dictionary-signal-point-codes
  FROM Classification

    ReportedAddress
  FROM Addresses

    Bunch,
    DataNetworkEquipment,
    IPAddress,
    IPPort,
    IPMask,
    NetworkPeerInfo,
    NetworkType
  FROM NetworkIdentifiers

    GeoLocation FROM Locations;

-- идентификатор оператора связи или структурного подразделения
TelcoID ::= INTEGER (0 .. 65535)

--- список идентификаторов операторов связи или структурного
подразделения
TelcoList ::= SEQUENCE OF TelcoID

-- запрос
```

```
DictionaryTask ::= SEQUENCE {
  id TAGGED.&id ({DictionaryTaskVariants}),
  data TAGGED.&Data ({DictionaryTaskVariants}{@id})
}

DictionaryTaskVariants TAGGED ::= { dictionaryTask }

dictionaryTask TAGGED ::= {
  OID { sorm-request-dictionaries }
  DATA ObjectDescriptor                    -- тип запрашиваемого
справочника (идентификатор отчёта)
}

-- ObjectDescriptor принимает значение одно из:
-- sorm-report-dictionary-telcos
-- sorm-report-dictionary-bunches
-- sorm-report-dictionary-basic-stations
-- sorm-report-dictionary-roaming-partners
-- sorm-report-dictionary-switches
-- sorm-report-dictionary-gates
-- sorm-report-dictionary-call-types
-- sorm-report-dictionary-supplement-services
-- sorm-report-dictionary-pay-types
-- sorm-report-dictionary-termination-causes
-- sorm-report-dictionary-ip-numbering-plan
-- sorm-report-dictionary-phone-numbering-plan
-- sorm-report-dictionary-doc-types
-- sorm-report-dictionary-ip-data-points
-- sorm-report-dictionary-special-numbers
-- sorm-report-dictionary-bunches-map
-- sorm-report-dictionary-mobile-subscriber-identity-plan
-- sorm-report-dictionary-signal-point-codes

-- Отчёт
DictionaryReport ::= SEQUENCE {
  id    TAGGED.&id ({DictionaryRecordsVariants}),   --- идентификатор
записи справочника
  data TAGGED.&Data({DictionaryRecordsVariants}{@id}) --- данные
записи справочника
}

DictionaryRecordsVariantsTAGGED ::= {
telcosRecords                    --- операторы связи,
обслуживаемые ИС ОРМ
  | bunchesRecords               --- пучки соединительных линий
  | basicStationsSectorRecords   --- базовые станции
```

```
        | roamingPartnersRecords                --- роуминговые партнеры
        | switchesRecords                       --- коммутаторы
        | gatesRecords                          --- IP шлюзы
        | callTypesRecords                      --- типы вызовов
        | supplementServicesRecords            --- список дополнительных видов
обслуживания (далее - ДВО)
        | payTypesRecords                       --- способы оплаты (пополнения
баланса)
        | terminationCausesRecords             --- причины завершения
соединения
        | ipNumberingPlanRecords               --- IP-план адресации
        | telephoneNumberingPlanRecords        --- план телефонной номерной
емкости
        | docTypesRecords                      --- типы документов,
удостоверяющих личность
        | ipDataPointsRecords                  --- идентификаторы точек
подключения к сети передачи данных, от которых получены записи о
соединениях
        | specialNumbersRecords                --- специальные номера
оператора (SMS-центры, ТМС, сервисы)
        | bunchesMapRecords                    --- карта связей пучков
соединительных линий
        | mobileSubscriberIdenityPlanRecords   --- план нумерации
идентификаторов мобильных телефонных абонентов
        | signalPointCodes--- коды сигнальных точек OPC/DPC
        }

        --- операторы связи, обслуживаемые ИС ОРМ
        telcosRecordsTAGGED ::= {
        OID { sorm-report-dictionary-telcos }
         DATA SEQUENCE OF TelcosRecord }

        TelcosRecord ::= SEQUENCE {
        telco-id          TelcoID,                    ---          номер
структурного подразделения или оператора связи
        begin-time   DateAndTime,                 --- время начала действия
        end-time     DateAndTime OPTIONAL,        ---  время  завершения
действия
        description  UTF8String (SIZE (1 .. 256)),  ---          описание
(наименование) оператора связи или структурного подразделения
        mcc [0]      NumericString (SIZE(3)) OPTIONAL,    --- код страны
        mnc [1]      NumericString (SIZE(3)) OPTIONAL     ---  код  оператора
связи
        }

        --- пучки соединительных линий
```

```
bunchesRecordsTAGGED ::= {
OID { sorm-report-dictionary-bunches }
 DATA SEQUENCE OF BunchRecord
}

BunchRecord ::= SEQUENCE {
telco-id          TelcoID,                    --- идентификатор
оператора связи или структурного подразделения
bunch-id    Bunch,                            --- идентификатор пучка
switch-id   UTF8String (SIZE (1 .. 128)),     --- идентификатор
коммутатора
bunch-type ENUMERATED {                       --- тип:
входящий/исходящий
inbound (0),
outbound (1),
bidirectional (3)
 },
begin-time  DateAndTime,                      --- время начала
назначения пучка
end-time    DateAndTime OPTIONAL,             --- время завершения
назначения пучка
description  UTF8String (SIZE(1 .. 256))      --- расшифровка пучка
}


--- базовые станции
basicStationsSectorRecords TAGGED ::= {
 OID { sorm-report-dictionary-basic-stations }
 DATA SEQUENCE OF BasicStationSectorRecord
}

BasicStationSectorRecord ::= SEQUENCE {
telco-id          TelcoID,                    --- идентификатор оператора
связи или структурного подразделения
begin-time  DateAndTime,                      --- время начала действия
базовой станции
end-time    DateAndTime,                      --- время завершения действия
базовой станции
address     UTF8String (SIZE (1 .. 256)),     --- произвольное текстовое
описание адреса или места расположения
sector-identifiers  BasicStationIdentifiers,  --- идентификаторы
сектора
antenna-configuration  BasicStationAntenna,   --- параметры антенной
системы
station-type      BasicStationType,           --- тип сети базовой
станции
```

```
        structured-address [0]    ReportedAddress OPTIONAL, ---    адрес    места
установки базовой станции
        location [1]        GeoLocationOPTIONAL              ---    географическое
местоположение
        }

        --- идентификаторы сектора
        BasicStationIdentifiers ::= CHOICE {
        telephone [0]       TelephoneIdentifiers,                    ---
идентификаторы сектора для телефонной сети
        wireless [1] SEQUENCEOFWirelessIdentifiers          ---    идентификаторы
сектора для сети передачи данных
        }

        --- идентификаторы сектора для телефонной сети
        TelephoneIdentifiers ::= SEQUENCE {
        lac    INTEGER (0 .. 65535),                        --- код зоны
        cell    INTEGER (0 .. 100000000000),                ---    идентификатор
сектора базовой станции (идентификатор базовой станции и сектор)
        cell-sign     UTF8String (SIZE (1 .. 18)) OPTIONAL    ---    телефонный
идентификатор соты
        }

        --- идентификаторы сектора для сети передачи данных
        WirelessIdentifiers ::=    SEQUENCE {
        cell    UTF8String (SIZE (1 .. 64)),                --- идентификатор сектора
        ip-list        IPListOPTIONAL,                      ---    перечень    назначенных
сектору IP-адресов/портов
        mac    OCTETSTRING (SIZE (6)) OPTIONAL    ---    MAC-адрес    сетевого
оборудования сектора
        }

        IPList ::= SEQUENCE OF NetworkPeerInfo

        --- параметры антенной системы
        BasicStationAntenna ::= CHOICE {
         gsm-antenna [0]     GsmAntenna,                     ---    параметры
антенной системы GSM-сектора
         cdma-antenna [1]     SEQUENCE OF CdmaAntenna,       ---    параметры
антенной системы CDMA-сектора
         wireless-antenna [2] SEQUENCE OF WirelessAntenna    ---    параметры
антенной системы WiFi/WiMAX-сектора
        }

        --- параметры антенной системы
        GsmAntenna ::= SEQUENCE {
```

```
        azimuthINTEGER (-1 .. 359),                --- азимут относительно направления
на север, в градусах, если значение равно «1», то направленность отсутствует
        widthINTEGER (0 .. 359),                   --- ширина растра в градусах
        horizon-angleINTEGER (0 .. 359),           --- угол наклона сектора к горизонту
        power [0]          INTEGER (0 .. 25000) OPTIONAL,          --- мощность
в ваттах (сектор)
        frequency [1]      INTEGER (0 .. 100000000000) OPTIONAL,   --- частота
излучения (сектор)
        vertical-lift [2]  INTEGER (0 .. 100) OPTIONAL,            --- высота подвеса
сектора
        gain-factor [3]    INTEGER (-100 .. 100) OPTIONAL,         --- коэффициент
усиления антенны (Дб)
        polarization [4]   INTEGER (-45 .. 45) OPTIONAL,           --- поляризация
антенной системы
        setting [5]        BsSetting OPTIONAL,                     ---          тип
расположения базовой станции
        thickness [6]      INTEGER (0 .. 359) OPTIONAL,            --- ширина диаграммы
направленности основного лепестка сектора базовой станции по вертикали
(в градусах)
        generation [7]     BsGeneration OPTIONAL,                  --- поколение базовой
станции
        controller-num [8] UTF8String (SIZE (1 .. 128)) OPTIONAL,  --- номер
контроллера базовой станции
        bcc-ncc [9]        UTF8String (SIZE (1 .. 128)) OPTIONAL,  --- код
идентификации базовой станции (BCC + NCC)
        cell-type [10]     BsCellTypeOPTIONAL,                     --- тип соты базовой
станции (макро-, микро-, пико-, фемто-)
        radiation-class [11] UTF8String (SIZE (1 .. 32)) OPTIONAL, --- класс
излучения базовой станции
        name [12]          UTF8String (SIZE (1 .. 32)) OPTIONAL,   --- наименование
базовой станции (номер базовой станции в базах данных оператора связи)
        channel [13]       UTF8String (SIZE (1 .. 32)) OPTIONAL    --- десятичный
номер канала
        }

        --- тип соты базовой станции (макро-, микро-, пико-, фемто-)
        BsCellType ::= ENUMERATED {
          macro (0),
          micro (1),
          pico (2),
          femto (3)
        }

        --- поколение базовой станции
        BsGeneration ::= ENUMERATED {
        g2 (0),
```

```
g3 (1),
g4 (2),
g5 (3)
}

--- тип расположения базовой станции
BsSetting ::= ENUMERATED {
 indoor (0),
 outdoor (1),
 underground (2)
}

CdmaAntenna ::= BroadbandWirelessParameters    --- параметры антенной системы CDMA-сектора
WirelessAntenna ::= BroadbandWirelessParameters    --- параметры антенной системы WiFi/WiMAX-сектора

BroadbandWirelessParameters ::=    SEQUENCE {
azimuth          INTEGER (-1 .. 359),          --- азимут относительно направления на север, в градусах, если значение равно «1», то направленность отсутствует
width            INTEGER (0 .. 359),           --- ширина растра в градусах
horizon-angle    INTEGER (0 .. 359),           --- угол наклона сектора к горизонту
power [0]        INTEGER (0 .. 25000) OPTIONAL,        --- мощность в ваттах (сектор)
frequency-start [1] INTEGER (0 .. 10000000000) OPTIONAL,--- нижняя частота излучения диапазона (сектор)
frequency-stop [2] INTEGER (0 .. 10000000000) OPTIONAL,--- верхняя частота излучения диапазона (сектор)
leaf-level [3]   INTEGER (-45 .. 45) OPTIONAL,  --- уровень боковых лепестков
vertical-lift [4]   INTEGER (0 .. 100) OPTIONAL,   --- высота подвеса сектора
gain-factor [5]   INTEGER (-100 .. 100) OPTIONAL, --- коэффициент усиления антенны (Дб)
polarization [6]   INTEGER (-45 .. 45) OPTIONAL   --- поляризация антенной системы
}

--- виды базовых станций
BasicStationType ::= ENUMERATED {
 gsm (0),
 cdma (1),
 umts (2),
```

```
    wifi (3),
    wimax (4)
  }

  roamingPartnersRecords TAGGED::= {
    OID { sorm-report-dictionary-roaming-partners }
    DATA SEQUENCE OF RoamingPartnerRecord
  }

  RoamingPartnerRecord ::= SEQUENCE {
  telco-id          TelcoID,                    --- идентификатор
оператора связи или структурного подразделения
  roaming-id  INTEGER (0 .. 4294967295),        --- идентификатор
роумингового партнёра
  begin-time  DateAndTime,                      --- время начала действия
роуминга
  end-time    DateAndTime OPTIONAL,             --- время завершения
действия роуминга
    description UTF8String (SIZE (1 .. 256))    --- описание
  }

  switchesRecords TAGGED ::= {
    OID { sorm-report-dictionary-switches }
    DATA SEQUENCE OF SwitchesRecord
  }

  SwitchesRecord ::= SEQUENCE {
  telco-id          TelcoID,                    --- идентификатор
оператора связи или структурного подразделения
  switch-id   UTF8String (SIZE (1 .. 128)),     --- идентификатор
коммутатора
  begin-time  DateAndTime,                      --- время начала действия
коммутатора
  end-time    DateAndTime OPTIONAL,             --- время завершения
действия коммутатора
  description  UTF8String (SIZE (1 .. 256)),    --- описание
  network-type      NetworkType,                --- тип сети связи
  address           ReportedAddress,            --- адрес места установки
коммутатора
  switch-sign NumericString (SIZE (1 .. 18)) OPTIONAL, --- телефонный
идентификатор коммутатора
  switch-type ENUMERATED {
  internal(0),      --- внутренний
  border(1)         --- пограничный
    }
  }
```

```
gatesRecords TAGGED ::= {
 OID { sorm-report-dictionary-gates }
 DATA SEQUENCE OF GatesRecord
}

GatesRecord ::= SEQUENCE {
telco-id          TelcoID,                    --- идентификатор
оператора связи или структурного подразделения
gate-id           INTEGER (0 .. 4294967295),  --- идентификатор
шлюза
ip-list      IPList,                          --- IP-адрес шлюза
begin-time   DateAndTime,                     --- время начала
действия шлюза
end-time     DateAndTime OPTIONAL,            --- время завершения
действия шлюза
description  UTF8String (SIZE (1 .. 256)),    --- описание
address           ReportedAddress,            --- адрес места
установки шлюза
gate-type ENUMERATED {                        --- тип IP-шлюза
sgsn(0),
ggsn(1),
smsc(2),
gmsc(3),
hss(4),
pstn(5),
voip-gw(6),
aaa(7),
nat(8)
 }
}

callTypesRecords TAGGED ::= {
 OID { sorm-report-dictionary-call-types }
 DATA SEQUENCE OF CallsTypesRecord
}

CallsTypesRecord ::= SEQUENCE {
telco-id          TelcoID,                    ---          идентификатор
оператора связи или структурного подразделения
call-type-id INTEGER  (0 .. 4294967295),      --- идентификатор типа
вызова
begin-time   DateAndTime,                     --- время начала действия
end-time     DateAndTime OPTIONAL,            --- время завершения
действия
description  UTF8String (SIZE (1 .. 256))     --- описание
```

```
}

supplementServicesRecords TAGGED ::= {
 OID { sorm-report-dictionary-supplement-services }
 DATA SEQUENCE OF SupplementServicesRecord
}

SupplementServicesRecord ::= SEQUENCE {
telco-id          TelcoID,                          --- идентификатор
оператора связи или структурного подразделения
service-id    INTEGER (0 .. 4294967295),            --- идентификатор
сервиса
mnemonic   UTF8String (SIZE (1..64)) OPTIONAL,      --- мнемоническое
обозначение сервиса
begin-time   DateAndTime,                           --- время начала
действия
end-time     DateAndTime OPTIONAL,                  --- время завершения
действия
 description UTF8String (SIZE (1 .. 256))           --- описание
}

payTypesRecords TAGGED ::= {
 OID { sorm-report-dictionary-pay-types }
 DATA SEQUENCE OF PayTypesRecord
}

PayTypesRecord ::= SEQUENCE {
telco-id          TelcoID,                          ----        идентификатор
оператора связи или структурного подразделения
pay-type-id  INTEGER (0 .. 4294967295),             --- идентификатор типа
оплаты
begin-time   DateAndTime,                           --- время начала действия
end-time     DateAndTime OPTIONAL,                  --- время завершения
действия
description  UTF8String (SIZE (1 .. 256))           --- описание
}

terminationCausesRecords TAGGED ::= {
 OID { sorm-report-dictionary-termination-causes }
 DATA SEQUENCE OF TerminationCausesRecord
}

TerminationCausesRecord ::= SEQUENCE {
 telco-id          TelcoID,                          --- идентификатор
оператора связи или структурного подразделения
```

```
        termination-cause-id    INTEGER (0 .. 16384),        --- код причины
        begin-time              DateAndTime,                 --- время начала действия
        end-time                DateAndTime OPTIONAL,        --- время завершения
действия
        description             UTF8String (SIZE (1 .. 256)),  --- описание
        network-type            NetworkType                  --- тип сети связи
       }

       ipNumberingPlanRecords TAGGED ::= {
        OID { sorm-report-dictionary-ip-numbering-plan }
        DATA SEQUENCE OF IpNumberingPlanRecord }

       IpNumberingPlanRecord ::= SEQUENCE {
        telco-id                TelcoID,                     --- идентификатор
оператора связи или филиала
        description UTF8String (SIZE (1 .. 256)),    --- описание назначения
диапазона
        network-address  IPAddress,                  --- подсеть
        network-mask     IPMask,                     --- маска подсети
        begin-time DateAndTime,                      --- время начала действия
        end-time    DateAndTime OPTIONAL             --- время завершения
действия
       }

       telephoneNumberingPlanRecords TAGGED ::= {
        OID { sorm-report-dictionary-phone-numbering-plan }
        DATA SEQUENCE OF TelephoneNumberingPlanRecord
       }

       TelephoneNumberingPlanRecord ::= SEQUENCE {
       telco-id                    TelcoID,                 --- идентификатор
оператора связи или структурного подразделения
       iso-3166-alpha-2            UTF8String (SIZE (2)),   --- 2-х символьная
аббревиатура страны
       iso-3166-alpha-3            UTF8String (SIZE (3)),   --- 3-х символьная
аббревиатура страны
       country-code                UTF8String (SIZE (3)),   --- международный
код страны
       national-significant-number UTF8String (SIZE (14)), --- номерной
телефонный префикс оператора связи, включая код зоны
       area-code-length            INTEGER (0 .. 6),        --- длина кода зоны в
телефонном префиксе оператора связи
       min-subscr-nr-length        INTEGER (1 .. 15),       --- минимальная
длина абонентского номера, символов (national-significant-number + min-subscr)
       max-subscr-nr-length        INTEGER (1 .. 15),       --- максимальная
длина абонентского номера, символов (national-significant-number + max-subscr)
```

```
        utc-min                 INTEGER (-12 .. 12),          --- минимальный
часовой пояс
        utc-max                 INTEGER (-12 .. 12),          --- максимальный
часовой пояс
        destination         UTF8String (SIZE (2 .. 256)),     --- страна
        operator-type-id        NetworkType,                  --- тип сети связи
оператора

        capacity-from           NumericString (SIZE (1 .. 15)),--- нижняя граница
диапазона выданных номеров (от)
        capacity-to         NumericString (SIZE (1 .. 15)) ,  --- верхняя граница
диапазона выданных номеров (до)
        capacity-size           INTEGER (1 .. 10000000),      --- количество
выделенных номеров в диапазоне (емкость)

        location                UTF8String (SIZE (0 .. 256)), --- текстовое
описание местоположения оператора связи
        registrar               UTF8String (SIZE (0 .. 256)), --- наименование
оператора связи
        range-activation        DateAndTime,                  --- дата и время
начала действия номерной емкости
        mobile-country-code     NumericString (SIZE(3)),      --- MCC
        mobile-network-code     NumericString (SIZE(3)),      --- MNC
        range-deactivation [0]  DateAndTimeOPTIONAL,          --- дата и время
завершения действия номерной емкости
        range-status [1]        UTF8String (SIZE (1 .. 128)) OPTIONAL, ---
текущее состояние номерной емкости
        description [2]         UTF8String (SIZE (1 .. 256))  OPTIONAL, ---
расшифровка оказываемых услуг связи по номерной емкости
        operating-company-number [3]  UTF8String (SIZE (0 .. 4)) OPTIONAL   ---
международный идентификатор оператора связи
        }

        docTypesRecords TAGGED ::= {
        OID {sorm-report-dictionary-doc-types}
        DATA SEQUENCE OF DocTypesRecord
        }

        DocTypesRecord ::= SEQUENCE {
        telco-id            TelcoID,                          --- идентификатор
оператора связи или структурного подразделения
        doc-type-id  INTEGER (0 .. 65535),                    --- идентификатор
типа документа
        begin-time   DateAndTime,                             --- время начала
действия
```

```
        end-time     DateAndTime OPTIONAL,                  --- время завершения
действия
        description  UTF8String (SIZE (1 .. 256))           ---          описание
(наименование)
        }

        ipDataPointsRecords TAGGED ::= {
        OID { sorm-report-dictionary-ip-data-points }
        DATA SEQUENCE OF IpDataPointRecord
        }

        IpDataPointRecord ::= SEQUENCE {
        telco-id             TelcoID,                       ---    идентификатор
оператора связи или структурного подразделения
        point-id             INTEGER (0 .. 1000),           ---    идентификатор
точки подключения
        description  UTF8String (SIZE (1 .. 256)),          ---          описание
(наименование) точки подключения
        begin-time   DateAndTime,                           ---    время    начала
действия
        end-time     DateAndTime OPTIONAL                   --- время завершения
действия
        }

        specialNumbersRecords TAGGED ::= {
         OID { sorm-report-dictionary-special-numbers }
         DATA SEQUENCE OF SpecialNumberRecord
        }

        SpecialNumberRecord ::= SEQUENCE {
        telco-id             TelcoID,                   ---          идентификатор
оператора связи или структурного подразделения
        directory-number  UTF8String (SIZE (1 .. 32)),  --- специальный номер
        description  UTF8String (SIZE (1 .. 256)),      ---                описание
(наименование, назначение) специального номера
        begin-time   DateAndTime,                       --- время начала действия
        end-time     DateAndTime OPTIONAL,              ---    время    завершения
действия
        network-address    IPAddress OPTIONAL           --- адрес в сети передачи
данных
        }

        bunchesMapRecords TAGGED ::= {
         OID { sorm-report-dictionary-bunches-map }
         DATA SEQUENCE OF BunchesMapRecord
        }
```

```
BunchesMapRecord ::= SEQUENCE {
a-bunch          BunchMapPoint,                    --- пара
коммутатор/пучок в связи
b-bunch          BunchMapPoint,                    --- пара
коммутатор/пучок в связи
begin-time  DateAndTime,                           --- время начала
действия связи
end-time    DateAndTime OPTIONAL                   --- время завершения
действия связи
}

BunchMapPoint ::= SEQUENCE {
 telco-id          TelcoID,                         --- идентификатор
оператора связи или филиала
 switch-id   UTF8String (SIZE (1 .. 128)),          --- идентификатор
коммутатора
 bunch-id    Bunch                                  --- идентификатор
пучка коммутатора
}

mobileSubscriberIdenityPlanRecords TAGGED ::= {
 OID { sorm-report-dictionary-mobile-subscriber-identity-plan }
 DATA SEQUENCE OF MobileSubscriberIdenityPlanRecord
}

MobileSubscriberIdenityPlanRecord ::= SEQUENCE {
 telco-id          TelcoID,                         --- идентификатор
оператора связи или структурного подразделения
 mcc               NumericString (SIZE(3)),         --- код страны
 mnc               NumericString (SIZE(3)),         --- код оператора связи
 area-code         NumericString (SIZE(3 .. 10)),   --- код зоны/региона
 capacity-from     NumericString (SIZE (0 .. 7)),   --- нижняя граница
диапазона (от)
 capacity-to       NumericString (SIZE (0 .. 7 )),  --- верхняя граница
диапазона (до)
 capacity-size     INTEGER (1 .. 10000000),         --- количество
выделенных в диапазоне (емкость)
 description       UTF8String (SIZE (2 .. 255)),    --- описание,
назначение
 region            UTF8String (SIZE (1 .. 128)),    --- область
 city        UTF8String (SIZE (1 .. 128)),   --- город, поселок, деревня
 range-activation       DateAndTime,                --- дата и время
начала действия номерной емкости
 range-deactivation [0]  DateAndTimeOPTIONAL,       --- дата и время
завершения действия номерной емкости
```

```
        range-status [1]        UTF8String (SIZE (2 .. 128)) OPTIONAL   ---
текущее состояние номерной емкости
        }

        -- Тип абонента
        PhoneAbonentType ::= ENUMERATED {
         local (0),                                  -- абонент данного коммутатора
         network (1),                                -- абонент сети связи
         roamer (2),                                 --  абонент  сети  подвижной
радиотелефонной связи, пользующийся услугами связи за пределами региона
регистрации (роумер)
         undefined (3)
        }

        --- коды сигнальных точек OPC/DPC
        signalPointCodes TAGGED ::= {
         OID {sorm-report-dictionary-signal-point-codes}
         DATA SEQUENCE OF SignalPointCodesRecord
        }

        SignalPointCodesRecord ::= SEQUENCE {
         ss7-point-code UTF8String (SIZE (1 .. 32)), --- идентификатор сигнальной
точки (OPC/DPC)
         switch-idUTF8String (SIZE (1 .. 128)),   --- идентификатор коммутатора
         begin-timeDateAndTime,                   --- время начала назначения пучка
         end-time       DateAndTime OPTIONAL,--- время завершения назначения
пучка
         description    UTF8String (SIZE (1 .. 256))  --- расшифровка пучка
        }

        END
```

4. Filters.asn

```
        Filters DEFINITIONS IMPLICIT TAGS ::=
        BEGIN

        EXPORTS filterMessage;

        IMPORTS TAGGED,
              sorm-message-filter
              FROM Classification

              IPAddress,
              IPPort,
              IPMask,
              PortRange
```

```
        FROM NetworkIdentifiers

        TelcoList
        FROM Dictionaries;

    filterMessage TAGGED ::= {
     OID sorm-message-filter
     DATA CHOICE {
      create-filter-request [0]  CreateFilterRequest,
      create-filter-response [1] CreateFilterResponse,
      drop-filter-request [2]    DropFilterRequest,
      drop-filter-response [3]   DropFilterResponse,
      get-filters-request [4]    GetFiltersRequest,
      get-filters-response [5]   GetFiltersResponse
     }
    }

    CreateFilterRequest ::= SEQUENCE {
     filter-id        FilterID,
     filter-parameters FilterParameters,
     allow-only-mode   BOOLEAN DEFAULT TRUE,
     telcos                    TelcoListOPTIONAL    --- список филиалов/узлов,
на которых устанавливается фильтр (для случаев, когда разные узлы сети
передачи данных реализованы в одних технических и программных средствах
ИС ОРМ с разными telco_id)
    }

    CreateFilterResponse ::= SEQUENCE {
     filter-id        FilterID,
     successful       BOOLEAN,
     error-description UTF8String (SIZE (1 .. 256)) OPTIONAL
    }

    DropFilterRequest ::= FilterID

    DropFilterResponse ::= SEQUENCE {
     filter-id        FilterID,
     successful       BOOLEAN,
     error-description UTF8String (SIZE (1 .. 256)) OPTIONAL
    }

    GetFiltersRequest ::= NULL

    GetFiltersResponse ::= SEQUENCE OF FilterResponse

    FilterParameters ::= SEQUENCE OF FilterParameter
```

```
FilterParameter ::= CHOICE {
  single-criteria [0] FilterSingleCriteria,   --- одиночный критерий
  pair-criteria [1]   FilterPairCriteria      --- два критерия объединенные по
логическому "И"
}

FilterPairCriteria ::= SEQUENCE {
  criteria-a [0] FilterSingleCriteria,   --- критерий 1
  criteria-b [1] FilterSingleCriteria    --- критерий 2
}

FilterSingleCriteria ::= CHOICE {
  ip-address [0]      IPAddress,                  --- IP-адрес
  ip-port [1]         IPPort,                     --- IP порт
  port-range [2]      PortRange,                  --- диапазон TCP/UDP портов
  vlan [3]            INTEGER (0 .. 4096),        --- 802.1Q метка (VLAN)
  mac [4]             OCTET STRING (SIZE (6)), --- MAC-адрес
  mpls-tag [5]        INTEGER,                    --- MPLS-метка
  sni [6]             UTF8String (SIZE (1 .. 128)), --- SSL/TLS server name
  http-content-type [7]  UTF8String (SIZE (1 .. 64)),  --- тип содержимого поля
Conten-typeHTTP-заголовка
  protocol-group [8]    INTEGER,                  --- группа прикладных
протоколов (значение 1 – torrent-протоколы передачи, 2 – протоколы потокового
медиаконтента, 3 – шифрованные протоколы)
  ip-protocol-number [9] INTEGER,                 --- номер интернет-протокола
по RFC1700
  http-cookie [10]     UTF8String,                  --- HTTP Cookie,
  http-uri [11]        UTF8String,                  --- HTTP URI
  ip-filter-mask [12]   IPFilterMask                --- IP маска
}

IPFilterMask ::= SEQUENCE {
  mask IPMask,              --- маска IP/сети
  mask-length INTEGER    --- длина маски
}

FilterResponse ::= SEQUENCE {
  filter-id        FilterID,
  filter-parameters FilterParameters,
  allow-only-mode   BOOLEAN,
  telcos                    TelcoListOPTIONAL          --- список филиалов,
на которых установлен конкретный фильтр
}

FilterID ::= INTEGER
```

```
END
```

5. Locations.asn

```
Locations DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS
   Location,
   GeoLocation;

IMPORTS
   NetworkPeerInfo FROM NetworkIdentifiers;


Location ::= CHOICE {
   mobile-location [0]     MobileLocation,          ---          местоположение
мобильного абонента
   wireless-location [1]   WirelessLocation,        ---          местоположение
абонента мобильной сети передачи данных
   geo-location [2]        GeoLocation,             ---          географическое
местоположение
   ip-location [3]         IpLocation               ---      ip местоположение
   }

MobileLocation ::= SEQUENCE {
   lac INTEGER (0 .. 65535),                        --- код зоны
   cellINTEGER (0 .. 100000000000),                 --- идентификатор базовой
станции
   ta [0]      INTEGER (0 .. 63) OPTIONAL,          ---      Timing      Advance
(временная компенсация)
   mcc [1]          NumericString (SIZE(3)) OPTIONAL,      --- код страны
   mnc [2]          NumericString (SIZE(3)) OPTIONAL       ---          код
оператора связи
   }

WirelessLocation ::= SEQUENCE {
cell  UTF8String (SIZE (1 .. 64)),                  --- идентификатор сектора
mac OCTETSTRING (SIZE (6))                          ---  MAC-адрес  сетевого
оборудования сектора
   }

GeoLocation ::= SEQUENCE {
latitude-grade      REAL,                                  --- широта в формате
[целое_число_градусов].[стотысячные_доли_градуса]
```

```
        longitude-grade    REAL,                              --- долгота в формате
[целое_число_градусов].[стотысячные_доли_градуса]
        projection-type    ENUMERATED {                       --- тип проекции
координат
          wgs84 (0),
          utm (1),
          sgs85 (2)
         }
        }

        IpLocation ::= NetworkPeerInfo

        END
```

6. Management.asn

```
        Management DEFINITIONS IMPLICIT TAGS ::=
        BEGIN

        EXPORTS managementMessage;

        IMPORTS TAGGED,
              sorm-message-management
              FROM Classification;

        managementMessage TAGGED ::= {
          OID { sorm-message-management }
          DATA CHOICE {
           request [0] ManagementRequest,
           response [1] ManagementResponse
        }
        }
        --- тип сообщения "команда управления ИС ОРМ"
        ManagementRequest ::= CHOICE {
         get-structure [0]          GetStructureRequest,      --- запрос на получение
структуры технических и программных средств ИС ОРМ: комплекса технических средств (далее – КТС) и модулей специального программного обеспечения (далее – СПО)
          get-module-config [1]  GetModuleConfigRequest, --- запрос на получение
конфигурации КТС/модуля СПО
          set-module-config [2]  SetModuleConfigRequest, --- запрос на изменение
конфигурации КТС/модуля СПО
          check-module [3]       CheckModuleRequest,       --- запрос на получение
состояния модуля
          get-module-types [4]   GetModuleTypesRequest   --- запрос на получение
типов модулей КТС и СПО
        }
```

```
--- запрос на получение структуры технических и программных средств ИС ОРМ конфигурации КТС и модулей СПО
GetStructureRequest ::= NULL

--- запрос на получение конфигурации КТС/модуля СПО
GetModuleConfigRequest::= CHOICE {
hw-modules-list [0]     RequestedHardwareModules,     --- перечень идентификаторов узлов КТС
sw-modules-list [1]     RequestedSoftwareModules      --- перечень идентификаторов модулей СПО
}

RequestedHardwareModules ::= SEQUENCEOFModuleId     --- перечень идентификаторов узлов КТС
RequestedSoftwareModules ::= SEQUENCEOFModuleId     --- перечень идентификаторов модулей СПО
--- запрос на изменение конфигурации КТС/модуля СПО
SetModuleConfigRequest::= SEQUENCE {
module-id   ModuleId,                                --- идентификатор конфигурируемого модуля
module-config     ConfiguratedModule                 --- устанавливаемая в модуль конфигурация
}

ConfiguratedModule ::= CHOICE {
 sw-module [0]    SormSoftwareModule,                --- для узла СПО
 hw-module [1]    SormHardwareModule                 --- для узла КТС
}

--- запрос на получение состояния модуля
CheckModuleRequest::= RequestedModulesList

RequestedModulesList ::= CHOICE {
 hw-modules [0]  RequestedHardwareModules,  --- идентификаторы узлов КТС, для которых запрашивается состояние
 sw-modules [1]  RequestedSoftwareModules   --- идентификаторы модулей СПО, для которых запрашивается состояние
}

--- запрос на получение типов модулей КТС и СПО
GetModuleTypesRequest ::= NULL

--- уникальный идентификатор КТС/модуля СПО
ModuleId ::= OCTET STRING (SIZE (8))
```

```
--- параметр модуля
ModuleParameter ::= SEQUENCE {
 parameter-name  UTF8String (SIZE (1 .. 256)),      --- наименование
параметра
read-only    BOOLEAN,                  --- контролируемый или
измеряемый параметр
parameter-value    ParameterValue              --- значение параметра
}

--- варианты значений параметров
ParameterValue ::= CHOICE {
string [0]    UTF8String (SIZE (1 .. 256)),
 integer [1] INTEGER (0 .. 999999999),
 boolean [2]       BOOLEAN
}

ModuleParameters ::= SEQUENCE OF ModuleParameter

SormSoftwareModule ::= SEQUENCE {
 module-id         ModuleId,         --- уникальный идентификатор
данного модуля
hardware-module-id      ModuleId,          --- идентификатор КТС,
на котором функционирует данный блок модуля СПО
block-name        INTEGER (0 .. 1024),      --- номер блока СПО
модуля
module-name       UTF8String (SIZE (1 .. 512)), --- наименование модуля
module-type       INTEGER (1 .. 512),       --- идентификатор типа
модуля
module-parameters       ModuleParameters,      --- список параметров
модуля
sub-modules-list  SubmodulesList OPTIONAL   --- субмодули
}

SubmodulesList ::= SEQUENCE OF SormSoftwareModule

SormHardwareModule ::= SEQUENCE {
 module-id         ModuleId,                  --- уникальный
идентификатор данного модуля
 block-name              INTEGER (0 .. 1024),       --- номер блока КТС
 module-name             UTF8String (SIZE (1 .. 512)), --- наименование
модуля
 module-parameters       HwParameterGroups          --- значение группы
параметров КТС
}

HwParameterGroup ::= SEQUENCE {
```

```
        group-name          UTF8String (SIZE (1 .. 512)),          ---          наименование
группы параметров для КТС
         module-parameters      ModuleParameters                  ---          перечень
параметров для КТС
        }

        HwParameterGroups ::= SEQUENCE OF HwParameterGroup

        SormSoftwareModules ::= SEQUENCE OF SormSoftwareModule

        SormHardwareModules ::= SEQUENCE OF SormHardwareModule

        --- тип сообщения "ответ на команду управления ИС ОРМ"
        ManagementResponse ::= CHOICE {
         get-structure [0]          GetStructureResponse,          --- ответ на запрос
получения структуры технических и программных средств ИС ОРМ: КТС и
модулей СПО
         get-module-config [1]   GetModuleConfigResponse,    --- ответ на запрос
получения конфигурации КТС/модуля СПО
         set-module-config [2]   SetModuleConfigResponse,    --- ответ на запрос
изменения конфигурации КТС/модуля СПО
         check-module [3]        CheckModuleResponse,         --- ответ на запрос
получения состояния модуля
         get-module-types [4]    GetModuleTypesResponse       --- ответ на запрос
получения типов модулей КТС и СПО
        }

        --- ответ на запрос получения структуры технических и программных
средств ИС ОРМ: КТС и модулей СПО
        GetStructureResponse ::= SEQUENCE {
         hw-modules      SormHardwareModules,          --- перечень всех
узлов КТС
         sw-modules      SormSoftwareModules           --- перечень всех
модулей СПО
        }

        --- ответ на запрос получения конфигурации КТС/модуля СПО
        GetModuleConfigResponse ::= SEQUENCE {
        hw-modules SormHardwareModules,          --- конфигурации
запрошенных узлов КТС
        sw-modules SormSoftwareModules           --- конфигурации
запрошенных модулей СПО
        }

        --- отчет на запрос изменения конфигурации КТС/модуля СПО
```

SetModuleConfigResponse::= ConfiguratedModule --- установленная в модуль конфигурация

--- ответ на запрос получения состояния модуля
CheckModuleResponse ::= CHOICE {
hw-modules [0] SormHardwareModules, --- текущее состояние запрошенных узлов КТС
sw-modules [1] SormSoftwareModules --- текущее состояние запрошенных модулей СПО
}

--- ответ на запрос получения типов модулей КТС и СПО
GetModuleTypesResponse ::= SEQUENCE OF ModuleType

ModuleType ::= SEQUENCE {
module-type INTEGER (1 .. 512), --- идентификатор типа модуля
type-description UTF8String (SIZE (1 .. 128)) --- расшифровка типа модуля
}

END

7. NetworkIdentifiers.asn

NetworkIdentifiers DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS Bunch, DataNetworkEquipment, IPAddress, IPPort, IPMask, NetworkPeerInfo, PortRange, NetworkType, DataVoipNumber, IMProtocol, VoipProtocol, HttpMethod, EntranceEventType;

--- идентификаторы пучка
Bunch ::= CHOICE {
gsm [0] UTF8String (SIZE (1 .. 32)), --- идентификатор пучка для GSM-Сети
cdma-umts [1] DataNetworkEquipment --- идентификатор пучка для W/CDMA, UMTS-сети
}

-- идентификатор оборудования сети передачи данных
DataNetworkEquipment ::= CHOICE {
mac [0] OCTETSTRING (SIZE (6)), --- MAC-адрес оконечного сетевого оборудования
atm [1] DataNetworkATM
}

```
--- ATM адрес (SDH/PDHсети)
DataNetworkATM ::= SEQUENCE {
vpi  OCTETSTRING (SIZE (1)),                    --- номер виртуального
пути сети ATM (VPI)
vci  OCTETSTRING (SIZE (2)) OPTIONAL    --- номер виртуального
канала сети ATM (VCI)
}

--- тип сети связи (стандарт)
NetworkType ::=  ENUMERATED {
not-specified(0),              --- не конкретизированный стандарт
mob-gsm(1),                    --- сеть подвижной радиотелефонной связи
стандарта GSM
mob-cdma(2),                   --- сеть подвижной радиотелефонной связи
стандарта CDMA
fix-pstn(3),                   --- сеть телефонной связи
data-ip(4),                    --- стационарные сети передачи данных
data-srv(5),                   --- TMC-службы
data-ip-mob(6),                --- передача данных в сети подвижной
радиотелефонной связи
data-ip-wifi (7),              --- беспроводная сеть передачи данных
стандарта WiFi
data-ip-max (8),               --- беспроводная сеть передачи данных
стандарта WiMAX
paging(9),                     --- персональный радиовызов
voip(10)                       --- передача голосовой информации в сети
передачи данных
}

--- IP-адрес
IPAddress ::= CHOICE {
 ipv4    [0]     IPV4Address,     --- IPv4-адрес
 ipv6    [1]     IPV6Address,     --- IPv6-адрес
 ip-range   [2]    IPRange                --- использование только в
RequestedConnections, RequestedIdentifiers, TaskLocation
}

--- IP-адрес без возможности указать диапазон значений адреса
IPAddressWithoutRange ::= CHOICE {
 ipv4    [0]     IPV4Address,             --- IPv4-адрес
 ipv6    [1]     IPV6Address              --- IPv6-адрес
}

-- IPv4-адрес
IPV4Address ::= OCTET STRING (SIZE (4))          --      IP-адрес
(4 байта)
```

```
-- IPv6-адрес
IPV6Address ::= OCTET STRING (SIZE (16))          -- IP-адрес (16 байт)

--- IP/UDP/TCP-порт
IPPort ::= OCTET STRING (SIZE (2)) -- порт

--- маска IP-подсети
IPMask ::= CHOICE {
 ipv4-mask [0]    IPV4Mask,                  --- IPv4-маска
 ipv6-mask [1]    IPV6Mask                   --- IPv6-маска
}

--- диапазон IP-адресов заданный маской или интервалом значений
IPRange ::= CHOICE {
 ip-address-range [0] IPAddressRange,
 ip-address-mask  [1] IPAddressMask
}

--- диапазон IP-адресов заданный интервалом значений
IPAddressRange ::= SEQUENCE {
 ip-from [0]IPAddressWithoutRange,   --- начало ip-интервала значений
 ip-to [1]   IPAddressWithoutRange   --- конец ip-интервала значений
}

--- диапазон IP-адресов заданный маской
IPAddressMask ::= SEQUENCE {
 ip-prefix IPAddressWithoutRange,  --- префикс IP/сети
 ip-mask IPMask                    --- маска IP/сети
}

--- информация об участнике соединения передачи данных
NetworkPeerInfo ::= SEQUENCE {
 ip-address  IPAddress,                  --- IP-адрес
 ip-port             IPPort OPTIONAL     --- IP-порт
}

PortRange ::= SEQUENCE {
 port-from [0]      IPPort,
 port-to [1]  IPPort
}

--- IPv4-маска
IPV4Mask ::= OCTET STRING (SIZE (4))

--- IPv6-маска
```

```
IPV6Mask ::= OCTET STRING (SIZE (16))

-- Номер телефона VoIP-абонента
DataVoipNumber ::= SEQUENCE {
 original-number          UTF8String (SIZE (1 .. 2048))OPTIONAL,        --- исходный номер абонента
 translated-number [0]    UTF8String (SIZE (1 .. 32)) OPTIONAL,   --- номер абонента после преобразования
 e164-number [1]          UTF8String (SIZE (1 .. 15)) OPTIONAL    --- номер абонента в E164-сети
}

VoipProtocol ::= ENUMERATED {
  sip(0),
  h323(1),
  iax(2),
  skype(100)
}
IMProtocol ::= ENUMERATED {    --- протокол, при помощи которого отправлены сообщения
icq(0),
aol(1),
msn(2),
yahoo(3),
web-mail(4),          --- передача сообщения пользователем посредством почтового веб-сервера
skype(5),
irc(6),
jabber(7),
mra(8),
tencent(9),
airway (10),
mms(98),
sms(99),
vk(100),
facebook(101),
myspace(102),
twitter(103),
odnoklassniki(104),
moymir(105),
zello(106),
other(151),
whatsapp(152),
viber(153),
telegram(154),
rezerv5(155),
```

```
rezerv6(156),
rezerv7(157),
rezerv8(158),
rezerv9(159),
rezerv10(160),
rezerv11(161),
rezerv12(162),
rezerv13(163),
rezerv14(164),
rezerv15(165),
rezerv16(166),
rezerv17(167),
rezerv18(168),
rezerv19(169),
rezerv20(170)
}

--- http метод
HttpMethod ::= ENUMERATED {
get(0),
post(1),
put(2),
delete (3)
   }

--- Тип события входа в личный кабинет
EntranceEventType ::= ENUMERATED {
registration(0),          --- регистрация
logon(1),                 --- вход
logon-failure(2),         --- неудачный вход
logoff(3),                --- выход
add-service(4),           --- заказ услуги
update-service(5),        --- изменение услуги
del-service(6),           --- прекращение услуги
add-domain(7),            --- подключение/покупка домена
del-domain(8),            --- удаление домена
create-support-ticket(9), --- создание запроса в службу поддержки
update-profile(10),       --- обновления информации в профиле
other(11)                 --- иное
}

END
```

8. ReportedIdentifiers.asn

```
ReportedIdentifiers DEFINITIONS IMPLICIT TAGS ::=
BEGIN
```

```
EXPORTS
  ReportedIdentifier
  ;

IMPORTS TAGGED,
      sorm-report-identifier-pager,
      sorm-report-identifier-pstn,
      sorm-report-identifier-gsm,
      sorm-report-identifier-cdma,
      sorm-report-identifier-data-network,
      sorm-report-identifier-voip
      FROM Classification

      IPAddress, DataNetworkEquipment, IPMask
      FROM NetworkIdentifiers;

ReportedIdentifier ::= SEQUENCE {
 id   TAGGED.&id ({ReportedIdentifierVariants}),
 data TAGGED.&Data ({ReportedIdentifierVariants}{@id})
}

--- варианты идентификаторов отчёта
ReportedIdentifierVariants TAGGED ::= {
   reportedPagerIdentifier                    --- идентификатор
сети персонального радиовызова
   | reportedPstnIdentifier                   --- идентификатор сети
телефонной связи
   | reportedGsmIdentifier                    --- идентификатор
GSM
   | reportedCdmaIdentifier                   --- идентификатор
CDMA
   | reportedDataNetworkIdentifier            --- идентификатор
сети передачи данных
   | reportedVoipIdentifier
}

--- идентификатор сети персонального радиовызова
reportedPagerIdentifier TAGGED ::= {
 OID { sorm-report-identifier-pager }
 DATA ReportedPagerIdentifier
}

ReportedPagerIdentifier ::= NumericString (SIZE (2 .. 18))

--- идентификатор телефонной сети общего пользования
```

```
reportedPstnIdentifierTAGGED ::= {
OID { sorm-report-identifier-pstn }
 DATA ReportedPstnIdentifier
}

ReportedPstnIdentifier ::= SEQUENCE {
directory-number UTF8String (SIZE (1 .. 32)),                    --- телефонный
номер в международном формате
internal-numberNumericString (SIZE (1 .. 32)) OPTIONAL    ---
дополнительный внутренний номер (при наличии)
}

-- идентификатор абонента GSM
reportedGsmIdentifier TAGGED ::= {
 OID { sorm-report-identifier-gsm }
 DATA ReportedGsmIdentifier
}

ReportedGsmIdentifier ::= SEQUENCE {
directory-number          UTF8String (SIZE (1 .. 32)),   ---      телефонный
номер в международном формате
 imsi                NumericString (SIZE (2 .. 18)),      ---   идентификатор
мобильного абонента
 imei [0] NumericString (SIZE (2 .. 18)) OPTIONAL,   ---   идентификатор
мобильной станции
 icc [1] NumericString (SIZE (10 .. 20)) OPTIONAL    ---   идентификатор
SIM-карты абонента
}

-- идентификатор абонента CDMA
reportedCdmaIdentifier TAGGED ::= {
 OID { sorm-report-identifier-cdma }
 DATA ReportedCdmaIdentifier
}

ReportedCdmaIdentifier ::= SEQUENCE {
 directory-number          UTF8String (SIZE (1 .. 32)),   ---      телефонный
номер в международном формате
 imsi          NumericString (SIZE (2 .. 18)),              ---   идентификатор
мобильного абонента
 esn [0] NumericString (SIZE (2 .. 18)) OPTIONAL, ---   идентификатор
мобильной станции
 min [1] NumericString (SIZE (2 .. 18)) OPTIONAL,    ---   идентификатор
мобильного абонента (CDMA)
 icc [2] NumericString (SIZE (10 .. 20)) OPTIONAL    ---   идентификатор
SIM-карты абонента
```

```
}

reportedDataNetworkIdentifier TAGGED ::= {
 OID { sorm-report-identifier-data-network }
 DATA ReportedDataNetworkIdentifier
}

ReportedDataNetworkIdentifier ::= SEQUENCE {
 user-equipment [0]      DataNetworkEquipment OPTIONAL,      ---
идентификатор пользовательского оборудования
 login [1]          UTF8String (SIZE (1 .. 128)) OPTIONAL, ---          имя
пользоавателя – login
 ip-address [2]          IPAddress OPTIONAL,                ---     IP-адрес
 e-mail [3]         UTF8String (SIZE (1 .. 256)) OPTIONAL, ---        адрес
электронной почты
 pin [4]     NumericString (SIZE (2 .. 20)) OPTIONAL,       ---        PIN
 phone-number [5]       UTF8String (SIZE (2 .. 32)) OPTIONAL,  --- номер
телефона
 user-domain [6]        UTF8String (SIZE (2 .. 256)) OPTIONAL, ---
пользовательский домен
 reserved [7]           UTF8String (SIZE (1 .. 256)) OPTIONAL, --- резерв
 ip-mask [8]            IPMask OPTIONAL
}

reportedVoipIdentifier TAGGED ::= {
 OID { sorm-report-identifier-voip }
 DATA ReportedVoipIdentifier
}

ReportedVoipIdentifier ::= SEQUENCE {
 ip-address [0]         IPAddress OPTIONAL,                ---     IP-адрес
абонента
 originator-name [1]    UTF8String (SIZE (1 .. 512)) OPTIONAL, ---
общедоступное имя инициатора связи
 calling-number [2]     UTF8String (SIZE (1 .. 32)) OPTIONAL   --- номер
вызывающего абонента
}

END
```

9. ReportsAbonents.asn

```
ReportsAbonents DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS AbonentsReport;
```

```
IMPORTS TAGGED,
        sorm-report-abonent-abonent,
        sorm-report-abonent-service,
        sorm-report-abonent-person,
        sorm-report-abonent-organization
    FROM Classification

        DateAndTime
    FROM Sorm

        TelcoID
    FROM Dictionaries

        ReportedIdentifier
    FROM ReportedIdentifiers

        NetworkType,
        NetworkPeerInfo,
        IPAddress
    FROM NetworkIdentifiers

        AddressType,
        ReportedAddresses,
        ReportedAddress
    FROM Addresses;


AbonentsReport ::= SEQUENCE {
 id   TAGGED.&id ({AbonentsReportVariants}),
 data TAGGED.&Data({AbonentsReportVariants}{@id})
}

AbonentsReportVariants TAGGED ::= {
 reportAbonent |                --- информация об абоненте
 reportService                  --- информация об сервисах
}

reportAbonent TAGGED ::= {
 OID sorm-report-abonent-abonent
 DATA SEQUENCE OF AbonentsRecord
}

AbonentsRecord ::= SEQUENCE {
telco-id            TelcoID,                          ---    идентификатор
оператора связи или структурного подразделения
```

```
        idents          ReportedIdentifier,                  --- идентификаторы
абонента
        contract-date   DateAndTime,                         --- дата и время
заключения договора
        contract        UTF8String (SIZE (1 .. 64)),         --- номер договора
        actual-from DateAndTime,                             --- дата и время
начала интервала времени, на котором актуальна информация по текущим
idents+abonent
        actual-to       DateAndTime,                         --- дата и время
окончания интервала времени, на котором актуальна информация по текущим
idents+abonent
        abonent         AbonentInfo,                         --- информация об
абоненте (клиенте оператора связи)
        status          ActiveStatus,                        --- текущий статус
абонента (подключен/отключен)
        attach [0]    DateAndTime OPTIONAL,                  --- дата и время
подключения услуги
        detach [1]    DateAndTime OPTIONAL,                  --- дата и время
отключения услуги
        services [3]  ActiveServices OPTIONAL,               --- активированные
услуги
        line-data [4]       LineData OPTIONAL,               --- линейные данные
(кросс, рамка, пара)
        standard [5] Standard OPTIONAL,                      --- стандарт связи
        addresses [6]       ReportedAddresses OPTIONAL,      --- адреса абонента
        last-ip [8]    NetworkPeerInfoOPTIONAL,              --- последний
IP-адрес абонента, зафиксированный при авторизации в системах IP-телефонии
        last-ip-date [9] DateAndTimeOPTIONAL                 --- дата фиксации
последнего IP-адреса абонента
        }

        ActiveServices ::= SEQUENCE OF AbonentService        --- набор
подключенных ДВО

        ActiveStatus ::= ENUMERATED {
          active (0),                                        --- активный
          not-active (1)                                     --- не активный
        }

        LineData ::= SEQUENCE {
          object [0]  UTF8String (SIZE (1 .. 128)) OPTIONAL, --- описание объекта
связи
          cross [1]   UTF8String (SIZE (1 .. 128)) OPTIONAL,  --- описание кросса
          block [2]   UTF8String (SIZE (1 .. 128)) OPTIONAL,  --- описние блока
          pair [3]          UTF8String (SIZE (1 .. 128)) OPTIONAL, ---     описание
пары
```

```
  reserved [4]      UTF8String (SIZE (1 .. 128)) OPTIONAL   --- резерв
 }

 AbonentInfo ::= SEQUENCE {
  id    TAGGED.&id ({AbonentsInfoVariants}),
  data   TAGGED.&Data({AbonentsInfoVariants}{@id})
 }

 AbonentsInfoVariants TAGGED ::= {
 abonentPerson |          --- информация об абоненте – физическом лице
 abonentOrganization      --- информация об абоненте – юридическом лице
 }

 --- Физическое лицо
 abonentPerson TAGGED ::= {
  OID { sorm-report-abonent-person }
  DATA AbonentPerson
 }

 AbonentPerson ::= SEQUENCE {
  name-info  PersonNameInfoReport,              --- фамилия, имя, отчество
(при наличии)
  birth-date  GeneralizedTime OPTIONAL,         --- дата рождения
  passport-info     PassportInfoReport,         --- паспортные данные
  bank [1]          UTF8String (SIZE(1 .. 256)) OPTIONAL,  ---          банк
абонента (используемый при расчетах с оператором связи)
  bank-account [2] UTF8String (SIZE(1 .. 30)) OPTIONAL,   ---          счет
абонента в банке (используемый при расчетах с оператором связи)
  e-mail [3]        UTF8String (SIZE (1 .. 256)) OPTIONAL,  ---        адрес
электронной почты
  phone-contact [4] UTF8String (SIZE (1 .. 128)) OPTIONAL, ---   телефоны
для взаимодействия
  ssid [5]          UTF8String (SIZE (1 .. 256)) OPTIONAL,    --- SSID
абонентской WiFI-точки доступа
  mac [6]           OCTET STRING (SIZE (6)) OPTIONAL   --- MAC-адрес
абонентской WiFI-точки доступа
 }

 PersonNameInfoReport ::= CHOICE {
  struct-name[0]    PersonStructNameInfoReport,       ---
структурированные фамилия, имя, отчество (при наличии)
  unstruct-name[1] UTF8String(SIZE(1 .. 1024))        ---
неструктурированные фамилия, имя, отчество (при наличии)
 }

 PersonStructNameInfoReport ::= SEQUENCE {
```

```
        given-name      UTF8String (SIZE (1 .. 256)),      --- имя
        initial         UTF8String (SIZE (1 .. 256)),      --- отчество (при
наличии)
        family-name     UTF8String (SIZE (1 .. 256))       --- фамилия
        }

        PassportInfoReport::= SEQUENCE {
        ident-card-info  IdentCardInfoReport,              --- описание
удостоверения личности
        doc-type-id      INTEGER (0 .. 65535)              --- идентификатор
типа документа, удостоверяющего личность
        }

        IdentCardInfoReport ::= CHOICE {
        struct-info [0]    IdentCardStructInfoReport,      ---
структурированная информация
        unstruct-info [1]  UTF8String (SIZE (1 .. 1024))   ---
неструктурированная информация
        }

        IdentCardStructInfoReport ::= SEQUENCE {
        ident-card-serial       UTF8String (SIZE (1..16)), --- серия
удостоверения личности
        ident-card-number       UTF8String (SIZE (1..16)), --- номер
удостоверения личности
        ident-card-description  UTF8String (SIZE (1 .. 512)) --- когда и кем
выдано
        }

        abonentOrganization TAGGED ::= {
        OID { sorm-report-abonent-organization }
        DATA AbonentOrganization
        }

        AbonentOrganization ::= SEQUENCE {
        full-name   UTF8String (SIZE (1 .. 256)),      --- полное наименование
        inn   UTF8String (SIZE(1 .. 64)),              --- ИНН
        contact [0]  UTF8String (SIZE (1 .. 128)) OPTIONAL, --- лицо,
взаимодействующее с оператором связи по вопросам оказания услуг связи
        phone-fax [1]      UTF8String (SIZE (1 .. 128)) OPTIONAL, --- телефоны,
факс (при наличии) для взаимодействия абонента – юридического лица с
оператором связи по вопросам оказания услуг связи
        internal-users [2]  InternalUsers OPTIONAL,    --- список
внутренних пользователей, выдается только при заполнении аналогичного поля
в запросе формирования задачи на поиск идентификаторов абонентов или при
```

заполнении поля internal-number в запросе на принадлежность идентификаторов операторам связи
bank [3] UTF8String (SIZE(1 .. 256)) OPTIONAL, --- банк абонента (используемый при расчетах с оператором связи)
bank-account [4] UTF8String (SIZE(1 .. 30)) OPTIONAL, --- счет абонента в банке (используемый при расчетах с оператором связи)
e-mail [5] UTF8String (SIZE (1 .. 256)) OPTIONAL, --- адрес электронной почты
ssid [6] UTF8String (SIZE (1 .. 256)) OPTIONAL, --- SSID абонентской WiFI-точки доступа
mac [7] OCTET STRING (SIZE (6)) OPTIONAL --- MAC-адрес абонентской WiFI-точки доступа
}

InternalUsers ::= SEQUENCE OF InternalUsersRecord --- набор записей, описывающих внутренних пользователей абонента – юридического лица

InternalUsersRecord ::= SEQUENCE {
user-name UTF8String (SIZE (1 .. 64)), --- строка – описатель внутреннего пользователя
internal-number NumericString (SIZE (1 .. 64)) --- внутренний номер (при наличии)
}

Standard ::= NetworkType --- перечень стандартов связи

reportService TAGGED ::= {
OID sorm-report-abonent-service
DATA SEQUENCE OF AbonentService
}

AbonentService ::= SEQUENCE {
telco-id TelcoID, --- идентификатор оператора связи или структурного подразделения
service-id INTEGER (0 .. 4294967295), --- идентификатор услуги
idents ReportedIdentifier OPTIONAL, --- идентификаторы абонента
contract UTF8String (SIZE (1 .. 64)), --- номер договора
begin-time DateAndTime, --- дата и время начала оказания услуги
end-time DateAndTime, --- дата и время окончания оказания услуги
parameter UTF8String (SIZE(1..256)) OPTIONAL --- индивидуальные параметры настройки услуги абонента
}

END

10. Reports.asn

```
Reports DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS reportMessage,
      Acknowledgement;

IMPORTS TAGGED,
      sorm-message-report
      FROM Classification

      Message, MessageID, DateAndTime
      FROM Sorm

      TaskID
      FROM Tasks

      DictionaryReport
      FROM Dictionaries

      ConnectionsReport
      FROM ReportsConnections

      LocationReport
      FROM ReportsLocations

      PaymentsReport
      FROM ReportsPayments

      PresenseReport
      FROM ReportsPresense

      NonFormalizedReport
      FROM ReportsNonFormalized

      AbonentsReport
      FROM ReportsAbonents

    DataContentReport
      FROM ReportsDataContent;

reportMessage TAGGED ::= {
 OID { sorm-message-report }
 DATA CHOICE {
 report [0]   Report,                          --- тип сообщения "отчет"
```

```
        ack [1]          Acknowledgement    ---    тип    сообщения
"подтверждение"
        }
    }

    -- Блок данных сообщения типа "отчет"
    Report ::= SEQUENCE {
    request-id          MessageID,      ---      идентификатор      запроса,
запросивший отчёт
    task-id             TaskID,         ---      идентификатор      задачи,
сгенерировавшей данный отчет
    total-blocks-number INTEGER, --- общее количество блоков в отчете
    block-number        INTEGER, --- порядковый номер текущего блока
    report-block        ReportDataBlock --- блок данных отчета
    }

    -- Подтверждение приёма блока передаётся с номером сообщения
соответствующему номеру сообщения блока отчёта
    Acknowledgement ::= SEQUENCE {
    successful BOOLEAN,                  --- признак корректного приёма
блока
    broken-record INTEGEROPTIONAL, --- номер записи в отчете,
обработанной на ПУ с ошибкой
    error-description UTF8String (SIZE(1..1024)) OPTIONAL --- описание
ошибки приёма блока в произвольной форме
    }

    ReportDataBlock ::= CHOICE {
    dictionary [0]          DictionaryReport,          --- отчеты задач
пополнения справочников (нормативно-справочная информация)
    abonents [1]        AbonentsReport,          --- отчеты задач поисков по
принадлежности абонентов
    connections [2]         ConnectionsReport,      --- отчеты задач поисков по
соединениям абонентов
    locations [3]       LocationReport,          --- отчет задачи получения
данных о местоположении абонентов
    payments [4]            PaymentsReport,         --- отчеты задач поисков по
совершенным платежам
    presense [6]        PresenseReport,          ---      отчеты      задач
по запросу наличия в ИС ОРМ информации
    nonFormalized [7] NonFormalizedReport,       ---      отчеты      задач
по обработке неформализованных данных
    data-content [10]       DataContentReport       ---      отчеты      задач      по
получению содержимого потоков
    }
```

```
END
```

11. ReportsConnections.asn

```
ReportsConnections DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS ConnectionsReport, CallsRecords;

IMPORTS  DateAndTime
  FROM Sorm

    DataContentID
  FROM Tasks

    Location
  FROM Locations

    ReportedIdentifier
  FROM ReportedIdentifiers

    Bunch,
    NetworkPeerInfo,
    DataNetworkEquipment,
    DataVoipNumber,
    IPAddress,
    IPMask,
    VoipProtocol,
    IMProtocol,
    HttpMethod,
    EntranceEventType
  FROM NetworkIdentifiers

    PhoneAbonentType,
    TelcoID
  FROM Dictionaries

    TAGGED,
    sorm-report-connection-pager,
    sorm-report-connection-pstn,
    sorm-report-connection-mobile,
    sorm-report-connection-ipdr-header,
    sorm-report-connection-aaa-login,
    sorm-report-connection-resource,
    sorm-report-connection-email,
    sorm-report-connection-im,
    sorm-report-connection-voip,
```

```
        sorm-report-connection-file-transfer,
        sorm-report-connection-term-access,
        sorm-report-connection-raw-flows,
        sorm-report-connection-address-translations,
        sorm-report-connection-entrance,
        sorm-report-connection-sms
    FROM Classification;

ConnectionsReport ::= CallsRecords

CallsRecords ::= SEQUENCE {
 id   TAGGED.&id ({ReportedCallsVariants}),
 data TAGGED.&Data ({ReportedCallsVariants}{@id})
}

ReportedCallsVariants TAGGED ::= {
   pagerRecord
 | pstnRecord
 | mobileRecord
 | dataAAARecord
 | dataEmailRecord
 | dataImRecord
 | dataFileTransferRecord
 | dataTermAccessRecord
 | dataRawFlowsRecord
 | dataResourceRecord
 | dataVoipRecord
 | dataAddressTranslationRecord
 | dataEntranceRecord
 | smsRecord
}

-- Детализированные записи отправки пэйджинг-сообщений
pagerRecord TAGGED ::= {
  OID { sorm-report-connection-pager }
  DATA SEQUENCE OF PagerRecordContent
}

PagerRecordContent ::= SEQUENCE {
telco-id          TelcoID,                    --- идентификатор
оператора связи или структурного подразделения
call-type-id INTEGER (0 .. 4294967295),       --- тип соединения
connection-time   DateAndTime,                --- дата и время передачи
сообщения
info   ReportedIdentifier,                    --- идентификаторы
абонента
```

```
    in-bytes-count      INTEGER (0 .. 1024),          --- объем переданных
данных
    term-cause  INTEGER (0 .. 16384)                  --- причина завершения
соединения
    }

    -- детализированные записи разговоров абонентов в сети телефонной
связи, в том числе неудавшиеся попытки соединений
    pstnRecord TAGGED ::= {
     OID { sorm-report-connection-pstn }
     DATA SEQUENCE OF PstnRecordContent
    }

    PstnRecordContent ::= SEQUENCE {
     telco-id                   TelcoID,                    --- идентификатор
оператора связи или структурного подразделения
     begin-connection-time  DateAndTime,                    --- дата и время
начала соединения
     duration                   INTEGER (0 .. 86399),       --- время соединения
     call-type-id               INTEGER  (0 .. 4294967295), --- тип соединения
     supplement-service-id  INTEGER (0 .. 4294967295),  --- ДВО при
соединении
     in-abonent-type        PhoneAbonentType,       --- тип вызывающего
абонента
     out-abonent-type       PhoneAbonentType,       --- тип вызываемого
абонента
     switch-id          UTF8String (SIZE (1 .. 128)), --- код коммутатора
обслужившего соединение
     inbound-bunch  UTF8String (SIZE (1 .. 32)),  --- входящий пучок
     outbound-bunch  UTF8String (SIZE (1 .. 32)),  --- исходящий пучок
     term-cause         INTEGER (0 .. 16384),       --- причина завершения
соединения
     phone-card-number [0] NumericString (SIZE (1.. 20)) OPTIONAL,--- номер
телефонной карты
     in-info [1]        ReportedIdentifier OPTIONAL,        ---
идентификаторы вызывающего абонента
     dialed-digits [2]          UTF8String (SIZE (1 .. 128)),       --- набранный
номер вызываемого абонента
     out-info [3]               ReportedIdentifier OPTIONAL,        ---
идентификаторы вызываемого абонента
     forwarding-identifier [4]      UTF8String (SIZE (2 .. 32)) OPTIONAL,  ---
телефонный номер при переадресации
     border-switch-id [5]   UTF8String (SIZE (1 .. 128)) OPTIONAL,  --- код
пограничного коммутатора
     message [10]               UTF8StringOPTIONAL,         --- текстовое
содержание сообщения абонента
```

```
        ss7-opc [11]            UTF8String (SIZE (1 .. 32)) OPTIONAL,  ---  SS7
код точки отправления
        ss7-dpc [12]            UTF8String (SIZE (1 .. 32)) OPTIONAL,  ---  SS7
код точки назначения
        data-content-id [13]    DataContentID OPTIONAL                 ---
идентификатор потока
        }

        -- детализированные записи разговоров абонентов сети подвижной
радиотелефонной связи, в том числе неудавшиеся попытки соединений
        -- детализированные записи об SMS сообщениях, в том числе неудавшихся
попытки отправки сообщений
        mobileRecord TAGGED ::= {
         OID { sorm-report-connection-mobile }
         DATA SEQUENCE OF MobileRecordContent
        }

        MobileRecordContent ::= SEQUENCE {
         telco-id                TelcoID,                    --- идентификатор
оператора связи или структурного подразделения
         begin-connection-time   DateAndTime,                --- дата и время
начала соединения
         duration                INTEGER (0 .. 86399),       --- время соединения
         call-type-id            INTEGER (0 .. 4294967295),  --- тип соединения
         supplement-service-id   INTEGER (0 .. 4294967295),  --- ДВО при
соединении
         in-abonent-type         PhoneAbonentType,           --- тип вызывающего
абонента
         out-abonent-type        PhoneAbonentType,           --- тип вызываемого
абонента
         switch-id          UTF8String (SIZE (1 .. 128)),    --- код коммутатора
обслужившего соединении
         term-cause         INTEGER (0 .. 16384),            --- причина
завершения соединения
         inbound-bunch [0]       Bunch OPTIONAL,             --- входящий пучок
         outbound-bunch [1]      Bunch OPTIONAL,             --- исходящий пучок
         in-info [2]        ReportedIdentifier OPTIONAL,     --- идентификаторы
вызывающего абонента
         in-end-location [3]     Location OPTIONAL,          --- местоположение
вызывающего абонента при завершении вызова
         in-begin-location [4]   LocationOPTIONAL,           --- местоположение
вызывающего абонента на начало вызова
         out-info [5]       ReportedIdentifier OPTIONAL,     --- идентификаторы
вызываемого абонента
         out-begin-location [6]  LocationOPTIONAL,           --- местоположение
вызываемого абонента на начало вызова
```

```
        out-end-location [7]     LocationOPTIONAL,          --- местоположение
вызываемого абонента при завершении вызова
        forwarding-identifier [8]      UTF8String (SIZE (2 .. 32)) OPTIONAL,  ---
телефонный номер при переадресации
        roaming-partner-id [9]  INTEGER (0 .. 4294967295) OPTIONAL,  ---    код
роумингового партнера
        border-switch-id [10]    UTF8String (SIZE (1 .. 128)) OPTIONAL,  ---    код
пограничного коммутатора
        data-content-id [41]     DataContentID OPTIONAL,                ---
идентификатор потока
        dialed-digits [42] UTF8String (SIZE (1 .. 128)) OPTIONAL        ---
набранный номер вызываемого абонента
    }

    -- детализированные записи об SMS-сообщениях, включающие текст
сообщений
    smsRecord TAGGED ::= {
     OID { sorm-report-connection-sms }
     DATA SEQUENCE OF SmsRecordContent
    }

    SmsRecordContent ::= SEQUENCE {
     telco-id                TelcoID,                    ---    идентификатор
оператора связи или структурного подразделения
     begin-connection-time  DateAndTime,                 ---    дата    и    время
начала соединения
     call-type-id            INTEGER (0 .. 4294967295),  --- тип соединения
     in-abonent-type         PhoneAbonentType,           --- тип вызывающего
абонента
     out-abonent-type        PhoneAbonentType,           --- тип вызываемого
абонента
     switch-id          UTF8String (SIZE (1 .. 128)),  ---    код    коммутатора,
обслужившего соединении
     term-cause         INTEGER (0 .. 16384),          ---    причина    завершения
соединения
     message            UTF8String,                    ---    текстовое    содержание
сообщения абонента
     inbound-bunch [0]       Bunch OPTIONAL,             --- входящий пучок
     outbound-bunch [1]      Bunch OPTIONAL,             --- исходящий пучок
     in-info [2]        ReportedIdentifier OPTIONAL,    ---    идентификаторы
вызывающего абонента
     in-end-location [3]     Location OPTIONAL,          ---    местоположение
вызывающего абонента при завершении вызова
     in-begin-location [4]   LocationOPTIONAL,           ---    местоположение
вызывающего абонента на начало вызова
```

```
        out-info [5]              ReportedIdentifier OPTIONAL,      ---
идентификаторы вызываемого абонента
        out-begin-location [6]    LocationOPTIONAL,          ---  местоположение
вызываемого абонента на начало вызова
        out-end-location [7]      LocationOPTIONAL,          ---  местоположение
вызываемого абонента при завершении вызова
        roaming-partner-id [9]    INTEGER (0 .. 4294967295) OPTIONAL, ---  код
роумингового партнера
        border-switch-id [10]     UTF8String (SIZE (1 .. 128)) OPTIONAL  ---  код
пограничного коммутатора
        }

        --- запись IPDR подключения/отключения абонента к сети связи
        dataAAARecord TAGGED ::= {
         OID { sorm-report-connection-aaa-login }
         DATA SEQUENCE OF DataAAARecordContent
        }

        DataAAARecordContent ::= SEQUENCE {
        telco-id                  TelcoID,                   ---  идентификатор
оператора связи или структурного подразделения
        point-id                  INTEGER (0 .. 1000),  --- идентификатор точки
подключения к сети передачи данных, от которой получена запись о соединении
        aaa-connection-time       DateAndTime,               ---  дата  и  время
подключения/отключения к сети передачи данных
        aaa-login-type            ENUMERATED {       --- тип соединения
        connect (0),                         ---  подключение  к  сети  передачи
данных
        disconnect (1),                      ---  отключение  от  сети  передачи
данных
        lac-update (2)
        },
        aaa-session-id      UTF8String (SIZE (1 .. 64)),       ---  идентификатор
сессии
        aaa-allocated-ip          IPAddress,                 ---  выделенный
динамический IP-адрес
        aaa-user-name             UTF8String (SIZE (1 .. 128)),      ---  имя
пользователя (login)
        aaa-connect-type          INTEGER (0 .. 65535),      ---  код
протокола в соответствии с RFC1700 либо номер порта для TCP/UDP
        aaa-calling-number        UTF8String (SIZE (2 .. 32)),       ---
вызывающий номер
        aaa-called-number UTF8String (SIZE (2 .. 32)),       ---  вызываемый
номер
        aaa-nas                   NetworkPeerInfo,           ---  IP-адрес/порт
NAS-сервера
```

```
        aaa-in-bytes-count INTEGER (0 .. 18446744073709551615),   ---   объем
принятых данных
        aaa-out-bytes-count      INTEGER (0 .. 18446744073709551615),  --- объем
переданных данных
        aaa-user-password [0]    UTF8String (SIZE (1 .. 128)) OPTIONAL,  ---
пользовательский пароль
        aaa-user-equipment [1]   DataNetworkEquipment OPTIONAL,          ---
идентификатор пользовательского оборудования
        aaa-apn [2]          UTF8String (SIZE (1 .. 128)) OPTIONAL,  ---
наименование точки доступа (Access Point Name)
        aaa-sgsn-ip [3]          IPAddress OPTIONAL,          ---     IP-адрес
GPRS/EDGE SGSN
        aaa-ggsn-ip [4]          IPAddress OPTIONAL,          ---     IP-адрес
GPRS/EDGE GGSN
        aaa-service-area-code    [5] INTEGER (0 .. 65535) OPTIONAL,   ---   код
зоны обслуживания (SAC) GPRS/EDGE
        aaa-location-start [6]   LocationOPTIONAL,            ---     базовая
станция, посредством которой установлено соединение (передача данных в сети
подвижной радиотелефонной связи)
         aaa-location-end [7]    LocationOPTIONAL,            ---     базовая
станция, посредством которой завершено соединение (передача данных в сети
подвижной радиотелефонной связи)
         phone-card-number [8] NumericString (SIZE (20)) OPTIONAL,   --- номер
телефонной карты
        aaa-imsi [9]         NumericString  (SIZE (2 .. 18)) OPTIONAL,    ---  IMSI
мобильного абонента
        aaa-imei [10]            NumericString (SIZE (2 .. 18)) OPTIONAL,    ---
идентификатор мобильной станции абонента
        aaa-esn [11]         NumericString (SIZE (2 .. 18)) OPTIONAL,    ---
идентификатор мобильной станции абонента
        aaa-pool-number [12]     UTF8String(SIZE (1 .. 64)) OPTIONAL,   --- номер
модемного пула
        aaa-mcc [13]             UTF8String OPTIONAL,
        aaa-mnc [14]             UTF8String OPTIONAL,
        aaa-allocated-ip-mask [15]      IPMask OPTIONAL
        }

        --- запись IPDR передачи почтового сообщения
        dataEmailRecordTAGGED ::= {
        OID { sorm-report-connection-email }
         DATA SEQUENCE OF DataEmailRecordContent
        }

        DataEmailRecordContent ::= CHOICE {
         mail-aaa [0]        DataEmailRecordContentAAA,
         mail-ipdr [1]       DataEmailRecordContentIPDR
```

```
}

DataEmailRecordContentIPDR ::= SEQUENCE {
  mail-cdr-header  DataNetworkCdrHeader,        ---                заголовок
IPDR-соединения
mail-event  EmailEvent,                  --- тип события
mail-sender UTF8String (SIZE (1 .. 512)), ---    отправитель    почтового
сообщения
mail-receiver     EmailReceivers,         ---    получатель    почтового
сообщения
mail-cc           EmailReceivers,         --- получатель-копия почтового
сообщения
mail-subject UTF8String (SIZE (0 .. 2048)), --- тема почтового сообщения
mail-size    INTEGER (0 .. 4294967295),   --- размер почтового сообщения,
включая прикрепленные файлы, байт
attachements      INTEGER (0 ..1),        ---    наличие    прикрепленных
файлов в письме (да/нет)
mail-servers EmailServers,                ---    список    текстовых
наименований почтовых серверов, посредством которых которые отправлено
письмо (сообщение), в том числе сервера веб-почты
mail-term-cause   INTEGER (0 .. 16384), ---    причина    завершения
соединения
mail-reply-to     UTF8String (SIZE (1 .. 256)) OPTIONAL, --- имя и адрес,
куда необходимо адресовать ответы на письмо
mail-protocol     ENUMERATED {                 ---    протокол,    при
помощи которого отправлено сообщение
    smtp(0),
pop3(1),
imap(2),
web-mail(3)
  } OPTIONAL,
mail-abonent-id [0]       UTF8String (SIZE (0 .. 64)) OPTIONAL,   ---
идентификатор абонента
mail-message [10] UTF8String OPTIONAL,                    ---    текстовое
содержание сообщения абонента
mail-nat-info [11]        SEQUENCE OF NetworkPeerInfo OPTIONAL,   ---
транслированные NAT IP/порт
mail-location [12]        Location OPTIONAL,   ---        местоположение
абонента
mail-data-content-id [13]DataContentID    OPTIONAL---    идентификатор
потока
}

DataEmailRecordContentAAA ::= SEQUENCE {
  mail-cdr-header         DataNetworkCdrHeader, ---              заголовок
IPDR-соединения
```

```
        mail-event          EmailEvent,                  --- тип события
        mail-aaa-info       IP-AAAInformation,           --- пользовательские
реквизиты входа
        mail-message [10]   UTF8StringOPTIONAL, --- текстовое содержание
сообщения абонента
        mail-nat-info [11] SEQUENCE OF NetworkPeerInfo OPTIONAL, ---
транслированные NAT IP/порт
        mail-location [12]          Location OPTIONAL,   --- местоположение
абонента
        mail-data-content-id [13]   DataContentID OPTIONAL   ---
идентификатор потока
    }

    EmailEvent ::= ENUMERATED
    {
        email-send(1),
        email-receive(2),
        email-download(3),
        email-logon-attempt(4),
        email-logon(5),
        email-logon-failure(6),
        email-logoff(7),
        email-partial-download(8)
    }

    EmailReceivers ::= SEQUENCE {
     data SEQUENCE OF UTF8String (SIZE (1 .. 512))
    }

    EmailServers ::= SEQUENCE {
     data SEQUENCE OF UTF8String (SIZE (1 .. 512))
    }

    --- запись IPDR передачи мгновенных электронных сообщений между
пользователями
    dataImRecord TAGGED ::= {
     OID { sorm-report-connection-im }
     DATA SEQUENCE OF DataImRecordContent
    }

    DataImRecordContent ::= SEQUENCE {
    im-cdr-header           DataNetworkCdrHeader,        ---        заголовок
IPDR-соединения
    im-user-login           UTF8String (SIZE (1 .. 128)), ---  учетная  запись
пользователя при подключении
```

```
        im-user-password        UTF8String (SIZE (1 .. 128)), --- пользовательский
пароль при подключении
        im-sender-screen-name   UTF8String (SIZE (1 .. 256)), --- общедоступное
имя отправителя
        im-sender-uin           UTF8String (SIZE (1 .. 256)), --- пользовательский
идентификатор отправителя (в том числе для чата сервера, предназначенного для
отправки мгновенных сообщений)
        im-receivers        ImReceivers,          --- список получателей
сообщения
        im-size                 INTEGER (0 .. 4294967295), --- размер данных
сессии, байт
        im-term-cause           INTEGER (0 .. 16384),      --- причина
завершения соединения

        im-protocol [0]         IMProtocol OPTIONAL,
        im-abonent-id [1]       UTF8String (SIZE (0 .. 64)) OPTIONAL, ---
идентификатор абонента
        im-event [5]        ImEvent OPTIONAL,
        im-message [10]         UTF8StringOPTIONAL,        --- текстовое
содержание сообщения абонента
         im-nat-info [11]       SEQUENCE OF NetworkPeerInfo OPTIONAL, ---
транслированные NAT IP/порт
        im-location [12]        Location OPTIONAL,         ---
местоположение абонента
        im-data-content-id [13]  DataContentID OPTIONAL            ---
идентификатор потока
        }

        ImReceivers ::= SEQUENCE OF ImReceiver

        ImReceiver ::= SEQUENCE {
        im-receiver-screen-name UTF8String (SIZE (1 .. 256)),      ---
общедоступное имя получателя
         im-receiver-uin        UTF8String (SIZE (1 .. 256))       ---
пользовательский идентификатор получателя (в том числе для сервера,
предназначенного для отправки мгновенных сообщений)
        }

        ImEvent ::= ENUMERATED
        {
            im-undefined(0),
            im-send(1),
            im-receive(2)
        }

        --- запись IPDR передачи файловых данных
```

```
dataFileTransferRecord TAGGED ::= {
 OID { sorm-report-connection-file-transfer }
 DATA SEQUENCE OF DataFileTransferRecordContent
}

DataFileTransferRecordContent ::= SEQUENCE {
file-cdr-header          DataNetworkCdrHeader,          --- заголовок
IPDR-соединения
file-server-name         UTF8String (SIZE (1 .. 256)),  --- название сервера
file-user-name           UTF8String (SIZE (0 .. 128)),  --- имя пользователя,
наименование учетной записи
file-user-password       UTF8String (SIZE (0 .. 256)),  --- пользовательский
пароль
file-server-type         BOOLEANOPTIONAL,               --- режим работы
файлового сервера (ACTIVE/PASSIVE)
file-in-bytes-count      INTEGER (0 .. 18446744073709551615), --- объем
принятых абонентом данных, (включает соединения управления
и соединения передачи данных), байт
file-out-bytes-count     INTEGER (0 .. 18446744073709551615), --- объем
переданных абонентом данных (включает соединения управления
и соединения передачи данных), байт
file-term-cause          INTEGER (0 .. 16384),          --- причина
завершения соединения
file-abonent-id [0]      UTF8String (SIZE (0 .. 64)) OPTIONAL, ---
идентификатор абонента
file-nat-info [10]       SEQUENCE OF NetworkPeerInfo OPTIONAL, ---
транслированные NAT IP/порт
file-location [11]       Location OPTIONAL,             ---
местоположение абонента
file-data-content-id [12] DataContentID OPTIONAL        ---
идентификатор потока
}

--- запись IPDR терминального доступа к оборудованию
dataTermAccessRecord TAGGED ::= {
 OID { sorm-report-connection-term-access }
 DATA SEQUENCE OF DataTermAccessRecordContent
}

DataTermAccessRecordContent ::= SEQUENCE {
 term-cdr-header         DataNetworkCdrHeader,          --- заголовок
IPDR-соединения
 term-in-bytes-count     INTEGER (0 .. 18446744073709551615), --- объем
принятых абонентом данных (включает соединения управления
и соединения передачи данных), байт
```

term-out-bytes-count INTEGER (0 .. 18446744073709551615), --- объем переданных абонентом данных (включает соединения управления и соединения передачи данных), байт
term-protocol ENUMERATED { --- протокол удаленного доступа к оборудованию
telnet(0),
ssh(1),
scp(2)
} OPTIONAL,
term-abonent-id [0] UTF8String (SIZE (0 .. 64)) OPTIONAL, --- идентификатор абонента
term-nat-info [10] SEQUENCE OF NetworkPeerInfo OPTIONAL, --- транслированные NAT IP/порт
term-location [11] Location OPTIONAL, --- местоположение абонента
term-data-content-id [12] DataContentID OPTIONAL, --- идентификатор потока
sni [13] UTF8String (SIZE (0 .. 128)) OPTIONAL --- SNI/CN
}

--- запись IPDR иных данных, принимаемых, получаемых пользователем при помощи закрытых протоколов обмена, а также DNS-запросы
dataRawFlowsRecord TAGGED ::= {
OID { sorm-report-connection-raw-flows }
DATA SEQUENCE OF DataRawFlowsRecordContent
}

DataRawFlowsRecordContent ::= SEQUENCE {
flow-cdr-header DataNetworkCdrHeader, --- заголовок IPDR-соединения
flow-in-bytes-count INTEGER (0 .. 18446744073709551615), --- объем принятых абонентом данных (включает соединения управления и соединения передачи данных), байт
flow-out-bytes-count INTEGER (0 .. 18446744073709551615), --- объем переданных абонентом данных (включает соединения управления и соединения передачи данных), байт
flow-protocol ENUMERATED { --- протокол передачи данных
ip(0),
udp(1),
tcp(2)
} OPTIONAL,
flow-abonent-id [0] UTF8String (SIZE (0 .. 64)) OPTIONAL, --- идентификатор абонента

```
        flow-nat-info [10]          SEQUENCE      OF      NetworkPeerInfo
OPTIONAL,       --- транслированные NAT IP/порт
        flow-location [11]          Location OPTIONAL,                  ---
местоположение абонента
        flow-data-content-id [12]   DataContentID OPTIONAL,             ---
идентификатор потока
        sni [13]                UTF8String (SIZE (0 .. 128)) OPTIONAL   ---
SNI/CN, DNS-query

    }

    --- запись IPDR HTTP-обращения к информационному ресурсу сети связи
(сайт, портал)
    dataResourceRecord TAGGED ::= {
     OID { sorm-report-connection-resource }
     DATA SEQUENCE OF DataResourceRecordContent
    }

    DataResourceRecordContent ::= SEQUENCE {
     res-cdr-header          DataNetworkCdrHeader,                      ---
заголовок IPDR-соединения
     res-url                 UTF8String (SIZE (1 .. 8192)),             ---
Наименование информационного ресурса
     res-bytes-count         INTEGER (0 .. 18446744073709551615),       ---
объем принятых и переданных данных в соединении (включает соединения
управления и соединения передачи данных), байт
     res-term-cause          INTEGER (0 .. 16384),                      ---
причина завершения соединения
     res-aaa-info [0]        IP-AAAInformationOPTIONAL,                 ---
пользовательские реквизиты входа в информационный ресурс
     res-http-method [1]     HttpMethod OPTIONAL,                       ---
http метод
     res-abonent-id [2]      UTF8String (SIZE (0 .. 64)) OPTIONAL,      ---
идентификатор абонента
     res-nat-info [10]       SEQUENCE OF NetworkPeerInfo OPTIONAL,      ---
транслированные NAT IP/порт
     res-location [11]       Location OPTIONAL,                         ---
местоположение абонента
     res-data-content-id [12] DataContentID OPTIONAL                    ---
идентификатор потока
    }

    --- запись IPDR голосовой связи посредством сети передачи данных, в том
числе с использованием Интернет-мессенджеров
    dataVoipRecord TAGGED ::= {
     OID { sorm-report-connection-voip }
```

```
    DATA SEQUENCE OF DataVoipRecordContent
}

DataVoipRecordContent ::= SEQUENCE {
    voip-cdr-header          DataNetworkCdrHeader,          --- заголовок
CDR-соединения
    voip-session-id          UTF8String (SIZE (0..64)),     --- идентификатор
сессии/call-id
    voip-conference-id       UTF8String (SIZE (1..64)),     --- идентификатор
конференции
    voip-duration                INTEGER (0 .. 864000),     --- длительность
разговора, секунд
    voip-originator-name     UTF8String (SIZE (1 .. 512)),  ---
общедоступное имя инициатора связи
    voip-call-type-id        INTEGER (0 .. 4294967295),     --- способ
подключения
    voip-calling-number      DataVoipNumber,                --- номер
вызывающего абонента (при вызовах с использованием Интернет-мессенджер
с использованием протокола STUN –внешние и внутренние сетевые реквизиты
абонента сети)
    voip-called-number       DataVoipNumber,                --- номер
вызываемого абонента (при вызовах с использованием Интернет-мессенджер
с использованием протокола STUN –внешние и внутренние сетевые реквизиты
удаленной стороны)
    voip-in-bytes-count      INTEGER (0 .. 18446744073709551615), --- объем
принятых абонентом данных (включает соединения управления
и соединения передачи данных), байт
    voip-out-bytes-count     INTEGER (0 .. 18446744073709551615), --- объем
переданных абонентом данных (включает соединения управления
и соединения передачи данных), байт
    voip-fax                     BOOLEAN,               --- наличие попытки
передачи факсовой информации (T.38)
    voip-term-cause          INTEGER (0 .. 16384),      --- причина
завершения соединения
    inbound-bunch [0]    Bunch OPTIONAL,                --- входящий пучок
    outbound-bunch [1]   Bunch OPTIONAL,                --- исходящий пучок
    voip-gateways [2]    SEQUENCE OF IPAddress OPTIONAL,    ---
идентификаторы медиашлюзов, обслуживших соединение
    voip-protocol [3] VoipProtocol OPTIONAL,
    supplement-service-id [4] INTEGER (0 .. 4294967295) OPTIONAL,
  --- ДВО при соединении
    voip-abonent-id [5]      UTF8String (SIZE (0 .. 64)) OPTIONAL,      ---
идентификатор абонента
    voip-nat-info [10] SEQUENCE OF NetworkPeerInfo OPTIONAL,  ---
транслированные NAT IP/порт
```

```
        voip-location [11]          Location OPTIONAL,   --- местоположение
абонента
        voip-event [12]         VoIPEvent OPTIONAL,        --- тип события
        voip-data-content-id [13]    DataContentID OPTIONAL --- идентификатор
потока
      }

      VoIPEvent ::= ENUMERATED {
       outgoing (0),          --- исходящее
       incoming (1),          --- входящее
       unknown (2)            --- направление неизвестно
      }

      --- запись IPDR трансляции сетевых адресов
      dataAddressTranslationRecord TAGGED ::= {
       OID { sorm-report-connection-address-translations }
       DATA SEQUENCE OF DataAddressTranslationRecordContent
      }

      DataAddressTranslationRecordContent ::= SEQUENCE {
        telco-id          TelcoID,                  ---          идентификатор
оператора связи или структурного подразделения
        point-id          INTEGER (0 .. 1000),      --- идентификатор точки
подключения к сети передачи данных, с которой получена запись о соединении
        translation-time  DateAndTime,              --- дата и время трансляции
        record-type       ENUMERATED {              ---  тип  записи  о
трансляции сетевого адреса
        session-start (0),                  --- начало сессии трансляции
        session-end (1)                     ---  окончание  сессии
трансляции
        },

        private-ip  NetworkPeerInfo,                --- внутренний адрес
        public-ip   NetworkPeerInfo,                --- внешний адрес
        destination-ip    NetworkPeerInfo,          --- адрес назначения
        translation-typeENUMERATED {                --- тип трансляции сетевых
адресов
            static-nat (0),                 --- статическая
            dynamic-nat (1),                    --- динамическая
            source-nat (2),                     --- источника
            destination-nat (3),                --- получателя
            pat (4)                         --- адрес-порт
        }
      }

      --- Data network header
```

```
DataNetworkCdrHeader ::= SEQUENCE {
  id  TAGGED.&id ({DataNetworkHeaderVariants}),
  data TAGGED.&Data ({DataNetworkHeaderVariants}{@id})
}

DataNetworkHeaderVariants TAGGED ::= { dataNetworkCdrHeader }

dataNetworkCdrHeader TAGGED ::= {
 OID { sorm-report-connection-ipdr-header }
 DATA DataNetworkCdrHeaderData
}

DataNetworkCdrHeaderData ::= SEQUENCE {
  telco-id                  TelcoID,                --- идентификатор
оператора связи или структурного подразделения
  begin-connection-time  DateAndTime,               --- дата и время начала
соединения
  end-connection-time    DateAndTime,               --- дата и время завершения
соединения
  client-info        NetworkPeerInfo,               --- информация о клиенте
(IP/порт)
  server-info        NetworkPeerInfo,               --- информация о сервере
(IP/порт)
  protocol-code          INTEGER (0 .. 65535),      --- код протокола
в соответствии с RFC1700 либо номер порта для TCP/UDP
  point-id    INTEGER (0 .. 1000) OPTIONAL   --- идентификатор точки
подключения к сети передачи данных, от которой получена запись о соединении
}

--- информация о входе в ресурс
IP-AAAInformation ::= SEQUENCE {
username    UTF8String (SIZE (0..64)),          --- пользовательский login
aaaResult   IP-AAAResultOPTIONAL                --- результат операции
входа
}

--- результат операции входа в ресурс
IP-AAAResult ::= ENUMERATED {
aaaUnknown(1),                                  --- результат неизвестен
aaaFailed(2),                                   --- неудачная попытка
входа
aaaSucceeded(3)                                 --- успешный вход
}

--- запись IPDR входа в личный кабинет
dataEntranceRecord TAGGED ::= {
```

```
    OID { sorm-report-connection-entrance }
    DATA SEQUENCE OF DataEntranceRecordContent
  }

  DataEntranceRecordContent ::= SEQUENCE {
    entrance-cdr-header DataNetworkCdrHeader,          --- заголовок IPDR-соединения
    entrance-event-type EntranceEventType,             --- тип события
    user-agent      [0] UTF8String (SIZE (1 .. 8192)) OPTIONAL,  --- заголовок User-agent HTTP соединения
    entrance-event-status [1] EntranceEventStatus OPTIONAL, --- статус события
    entrance-service-id [2] INTEGER (0 .. 4294967295) OPTIONAL, --- идентификатор заказанной услуги
    entrance-abonent-id [3] ReportedIdentifier OPTIONAL, --- идентификатор абонента (data-network-identifier)
    entrance-owner-id [4] ReportedIdentifier OPTIONAL, --- идентификатор владельца аккаунта, если не совпадает с идентификатором абонента (data-network-identifier)
    entrance-service-parameter [5] EntranceServiceParameter OPTIONAL,
    entrance-location [6] LocationOPTIONAL,    --- местоположение абонента (при наличии возможности определения)
    entrance-data-content-id [7]  DataContentID OPTIONAL          --- идентификатор потока
  }

  EntranceServiceParameter ::= SEQUENCE {
    ip-address [0]          IPAddress OPTIONAL,          --- IP-адрес
    domain [1]              UTF8String (SIZE (2 .. 256)) OPTIONAL, --- название подключенного/отключенного/удаленного домена
    e-mail [2]        UTF8String (SIZE (1 .. 256)) OPTIONAL, ---      адрес электронной почты
    parameter [3] UTF8String (SIZE (1 .. 8192)) OPTIONAL --- индивидуальные параметры настройки услуги абонента
  }

  --- Статус заказа/изменения/прекращения услуги в личном кабинете
  EntranceEventStatus ::= ENUMERATED {
  event-failed (0),      --- ошибка выполнения заказа/изменения/прекращения услуги, подключения/покупки/удаления домена, обновления информации в профиле
  event-succeeded (1) --- успешное выполнение заказа/изменения/прекращения услуги, подключения/покупки/удаления домена, обновления информации в профиле
  }
```

```
END
```

12. ReportsLocations.asn

```
ReportsLocations DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS LocationReport;

IMPORTS DateAndTime
  FROM Sorm

    TelcoID
  FROM Dictionaries

    Location
  FROM Locations

    ReportedIdentifier
  FROM ReportedIdentifiers;

LocationReport ::= SEQUENCE OF ValidateLocationRecord

ValidateLocationRecord ::= SEQUENCE {
    telco-id          TelcoID,              --- идентификатор оператора
связи или структурного подразделения
    connection-time   DateAndTime,          --- время определения
местоположения
    ident  ReportedIdentifier,              --- идентификатор мобильного
абонента
    connection-locationLocation             --- местоположение мобильного
абонента
    }

END
```

13. ReportsNonFormalized.asn

```
ReportsNonFormalized DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS NonFormalizedReport;

IMPORTS
    EntityId,
    NonFormalizedEntityAttributeData
    FROM TasksNonFormalized
```

```
        StandardInterval
        FROMReportsPresense;

    --- отчет задачи по обработке неформализованных данных
    NonFormalizedReport ::= CHOICE {
     nonformalized-report [0]      NonFormalizedRecords, --- отчет по задаче
обработки неформализованных данных
      nonformalized-presense [1]   NonFormalizedPresenseInfo  ---  отчет
по наличию неформализованных данных заданного вида
    }

    NonFormalizedRecords ::= SEQUENCE OF NonFormalizedRecord

    NonFormalizedPresenseInfo ::= SEQUENCE OF StandardInterval        ---
диапазоны времени за которые есть неформализованные данные

    --- запись неформализованных данных
    NonFormalizedRecord ::= SEQUENCE OF NonFormalizedEntityAttributeData

    END
```

14. ReportsPayments.asn

```
ReportsPayments DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS PaymentsReport;

IMPORTS TAGGED,
      sorm-report-payment-bank-transaction,
      sorm-report-payment-express-pays,
      sorm-report-payment-terminal-pays,
      sorm-report-payment-service-center,
      sorm-report-payment-cross-account,
      sorm-report-payment-telephone-card,
      sorm-report-payment-balance-fillups,
      sorm-report-payment-bank-division-transfer,
      sorm-report-payment-bank-card-transfer,
      sorm-report-payment-bank-account-transfer
   FROM Classification

      DateAndTime
   FROM Sorm

      TelcoID
   FROM Dictionaries
```

```
    Location
  FROM Locations

    ReportedIdentifier
  FROM ReportedIdentifiers

    ReportedAddress
  FROM Addresses;

PaymentsReport ::= SEQUENCE {
 id TAGGED.&id ({ReportedPaymentsVariants}),
 data TAGGED.&Data ({ReportedPaymentsVariants}{@id})
}

--- варианты запрашиваемых параметров связей
ReportedPaymentsVariantsTAGGED ::= {
bankTransactionReport
 | expressCardReport
 | publicTerminalReport
 | serviceCenterReport
 | crossAccountReport
 | telephoneCardReport
 | balanceFillupReport
 | bankDivisionTransferReport
 | bankCardTransferReport
 | bankAccountTransferReport
}

--- отчет задачи на поиск пополнения баланса посредством банковского
перевода
bankTransactionReport TAGGED ::= {
  OID { sorm-report-payment-bank-transaction }
  DATA SEQUENCE OF BankTransactionRecord
}

BankTransactionRecord ::= SEQUENCE {
telco-id          TelcoID,                  --- идентификатор
оператора связи или структурного подразделения
device-id    ReportedIdentifier,            --- идентификатор
абонента
date-time-fillup    DateAndTime,            --- время и дата
пополнения баланса
bank-account        UTF8String (SIZE (1 .. 64)), --- номер банковского
счета, с которого совершен платеж
bank-name  UTF8String (SIZE (1 .. 512)),    --- наименование банка, со
счета которого совершен перевод
```

```
        bank-address        ReportedAddress,                --- адрес банка, со счета
которого совершен перевод
        amount              UTF8String (SIZE (1 .. 64))     --- сумма перевода
        }

        --- отчет задачи на поиск пополнения баланса с использованием карты
экспресс-оплаты
        expressCardReport TAGGED ::= {
          OID { sorm-report-payment-express-pays }
          DATA SEQUENCE OF ExpressPaysRecord
        }

        ExpressPaysRecord ::= SEQUENCE {
        telco-id            TelcoID,                        ---          идентификатор
оператора связи или структурного подразделения
        device-id     ReportedIdentifier,                   ---          идентификатор
абонента
        date-time-fillup    DateAndTime,                    ---     время     и     дата
пополнения баланса
        card-number         UTF8String (SIZE (1 .. 64)),    --- номер карты
        amount              UTF8String (SIZE (1 .. 64))     --- сумма перевода
        }

        --- отчет задачи на поиск пополнения баланса с ипользованием терминала
моментальных платежей
        publicTerminalReport TAGGED ::= {
          OID { sorm-report-payment-terminal-pays }
          DATA SEQUENCE OF PublicTerminalRecord
        }

        PublicTerminalRecord ::= SEQUENCE {
          telco-id          TelcoID,                        ---          идентификатор
оператора связи или структурного подразделения
          device-id   ReportedIdentifier,                   ---          идентификатор
абонента
          date-time-fillup  DateAndTime,                    ---     время     и     дата
пополнения баланса
          terminal-id UTF8String (SIZE (1 .. 64)),    --- идентификатор терминала
          terminal-number  NumericString (SIZE (2 .. 20)),      --- номер терминала
          terminal-address  ReportedAddress,                --- адрес терминала
          amount            UTF8String (SIZE (1 .. 64)),    --- сумма перевода
          location    Location OPTIONAL                 --- адрес совершения платежа
        }

        --- отчет задачи на поиск пополнения баланса посредством центров
обслуживания клиентов
```

```
serviceCenterReport TAGGED ::= {
  OID { sorm-report-payment-service-center }
  DATA SEQUENCE OF ServiceCenterRecord
}

ServiceCenterRecord ::= SEQUENCE {
  telco-id          TelcoID,                    --- идентификатор
оператора связи или структурного подразделения
  device-id   ReportedIdentifier,               --- идентификатор
абонента
  date-time-fillup  DateAndTime,                --- время и дата
пополнения баланса
  center-id   UTF8String (SIZE (1 .. 64)),  --- идентификатор центра
обслуживания клиентов
  center-address    ReportedAddress,            --- адрес центра
обслуживания клиентов
  amount            UTF8String (SIZE (1 .. 64))   --- сумма перевода
}

--- отчет задачи на поиск пополнения баланса посредством снятия денег со
счета другого абонента
crossAccountReport TAGGED ::= {
  OID { sorm-report-payment-cross-account }
  DATA SEQUENCE OF CrossAccountRecord
}
CrossAccountRecord ::= SEQUENCE {
telco-id          TelcoID,                    --- идентификатор
оператора связи или структурного подразделения
device-id   ReportedIdentifier,               --- идентификатор
абонента-получателя платежа
date-time-fillup  DateAndTime,                --- время и дата
пополнения баланса
donanted-id ReportedIdentifier,               --- идентификатор
абонента-отправителя платежа
amount      UTF8String (SIZE (1 .. 64))       --- сумма перевода
}

--- отчет задачи на поиск пополнения баланса посредством телефонных
карт
telephoneCardReport TAGGED ::= {
  OID { sorm-report-payment-telephone-card }
  DATA SEQUENCE OF ValidateTelephoneCardRecord
}

ValidateTelephoneCardRecord ::=   SEQUENCE {
```

```
        telco-id                TelcoID,                    --- идентификатор
оператора связи или структурного подразделения
        activator-device-idReportedIdentifier,              --- идентификатор
абонента, активировавшего карту
        date-time-fillup        DateAndTime,                --- время и дата
пополнения баланса
        card-number             NumericString (SIZE (2 .. 20)),--- номер телефонной
карты
        amount                  UTF8String (SIZE (1 .. 64))  --- сумма перевода
        }

        --- отчет задачи на поиск пополнения баланса личного счета абонента
        balanceFillupReport TAGGED ::= {
        OID { sorm-report-payment-balance-fillups }
        DATA SEQUENCE OF ValidateBalanceFillupRecord
        }

        ValidateBalanceFillupRecord ::=     SEQUENCE {
        telco-id            TelcoID,                        --- идентификатор
оператора связи или структурного подразделения
        pay-type-id INTEGER (0 .. 4294967295),          --- тип платежа (способ
оплаты)
        device-id    ReportedIdentifier,                    --- идентификатор
абонента
        date-time-fillup    DateAndTime,                    --- время и дата
пополнения баланса
        amount              UTF8String (SIZE (1 .. 64)),    --- сумма перевода
        pay-parameters      UTF8String (SIZE (1 .. 512)) OPTIONAL   ---
«неструктурированная" информация
        }

        --- отчет задачи по переводу средств со счета абонента для их снятия
в отделении банка
        bankDivisionTransferReport TAGGED ::= {
          OID { sorm-report-payment-bank-division-transfer }
          DATA SEQUENCE OF ValidateBankDivisonTransferRecord
        }

        ValidateBankDivisonTransferRecord ::= SEQUENCE {
            telco-id                TelcoID,            ---         идентификатор
оператора связи или структурного подразделения
            donanted-id             ReportedIdentifier, ---         идентификатор
инициатора перевода средств
            date-time-fillup        DateAndTime,    --- время и дата перевода
средств
```

```
        person-recieved     UTF8String(SIZE (1 .. 512)), --- получатель
платежа (ФИО и прочая неструктурированная информация)
        bank-name           UTF8String (SIZE (1 .. 256)), --- наименование
банка получателя
        bank-division-name  UTF8String (SIZE (1 .. 512)), --- наименование
отделения банка получателя
        bank-division-address   ReportedAddress,        --- адрес отделения
банка получателя
        amount              UTF8String (SIZE (1 .. 64))  --- сумма перевода
    }

    --- отчет задачи по переводу средств со счета абонента на банковскую
карту
    bankCardTransferReport TAGGED ::= {
      OID { sorm-report-payment-bank-card-transfer }
      DATA SEQUENCE OF ValidateBankCardTransferRecord
    }

    ValidateBankCardTransferRecord ::= SEQUENCE {
        telco-id            TelcoID,                    --- идентификатор
оператора связи или структурного подразделения
        donanted-id  ReportedIdentifier,                --- идентификатор
инициатора перевода средств
        bank-card-id  NumericString (SIZE (1 .. 12)),   --- номер банковской
карты получателя перевода
        date-time-fillup    DateAndTime,                --- время и дата
перевода средств
        amount      UTF8String (SIZE (1 .. 64))         --- сумма перевода
    }

    --- отчет задачи по переводу средств со счета абонента на счет в банке
    bankAccountTransferReport TAGGED ::= {
      OID { sorm-report-payment-bank-account-transfer }
      DATA SEQUENCE OF ValidateBankAccountTransferRecord
    }

    ValidateBankAccountTransferRecord ::= SEQUENCE {
        telco-id            TelcoID,                    ---
идентификатор оператора связи или структурного подразделения
        donated-id   ReportedIdentifier,                ---
идентификатор инициатора перевода средств
        bank-name   UTF8String (SIZE (1 .. 256)), --- наименование банка
получателя
        bank-account UTF8String (SIZE (1 .. 64)), --- номер банковского
счета получателя
```

```
        date-time-fillup      DateAndTime,              --- время и дата перевода
средств
        amount        UTF8String (SIZE (1 .. 64))     --- сумма перевода
    }

    END
```

15. ReportsPresense.asn

```
ReportsPresense DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS PresenseReport,
    StandardInterval;

IMPORTS FindRange,
    DateAndTime
  FROM Sorm

    TelcoID
  FROM Dictionaries

    Location
  FROM Locations

    ReportedIdentifier
  FROM ReportedIdentifiers

    NetworkType
  FROM NetworkIdentifiers

    TAGGED,
    sorm-report-presense-abonents,
    sorm-report-presense-connections,
    sorm-report-presense-payments,
    sorm-report-presense-dictionaries,
    sorm-report-presense-locations
  FROMClassification;

---отчет по запросу наличия информации
PresenseReport ::= SEQUENCE {
 id   TAGGED.&id ({ReportedPresensesVariants}),
 data TAGGED.&Data ({ReportedPresensesVariants}{@id})
}

ReportedPresensesVariants TAGGED ::= {
  subsPresence
```

```
  | connectionsPresence
  | paymentsPresense
  | dictionariesPresence
| locationPresence
}
```

--- отчет по наличию информации по абонентам и их идентификаторам. Для каждого стандарта указывается более одного или ни одного интервала (фактические периоды наличия информации)

```
subsPresence TAGGED ::= {
  OID { sorm-report-presense-abonents }
  DATASEQUENCEOFStandardInterval
}
```

--- отчет по наличию информации по соединениям. Для каждого стандарта указывается более одного или ни одного интервала (фактические периоды наличия информации);

```
connectionsPresence TAGGED ::= {
  OID { sorm-report-presense-connections }
  DATA SEQUENCE OF ConnectionsPresenseRecord
}

ConnectionsPresenseRecord ::= SEQUENCE {
standard-interval StandardInterval,
data-type ENUMERATED {  --- вид соединений передачи данных, информация по которым есть в ИС ОРМ
    telephone-pstn (0),          --- соединения телефонной сети связи
    telephone-mobile (1),        --- телефонные соединения сети подвижной радиотелефонной связи
    pager (2),                   --- соединения сети персонального радиовызова
    data-aaa (3),                --- подключения/отключения абонента к сети связи
    data-resource (4),           --- HTTP-обращения к информационному ресурсу сети связи (сайт, портал)
    data-email (5),              --- передача почтовых сообщений
    data-im (6),                 --- передача мгновенных электронных сообщений между пользователями
    data-voip (7),               --- голосовая связь посредством сети передачи данных
    data-file (8),               --- передача файловых данных
    data-term (9),               --- терминальный доступ к оборудованию
    data-raw (10),               --- иные данные, принимаемые, получаемые пользователем при помощи закрытых протоколов обмена
    data-address-translations (11), --- трансляции сетевых адресов
    data-entrance (12)           --- действие в личном кабинете
```

```
}
}

paymentsPresense TAGGED ::= {
  OID { sorm-report-presense-payments }
  DATA SEQUENCE OF StandardInterval --- описание имеющейся информации по пополнениям балансов абонентов
}

--- отчет о наличии информации справочников в ИС ОРМ. Если любой из справочников не публикуется ИС ОРМ, запись о нем отсутствует
  dictionariesPresence TAGGED ::= {
  OID { sorm-report-presense-dictionaries }
  DATA SEQUENCE OF DictionaryInfo
}

--- запись отчета о наличии информации справочников
DictionaryInfo ::= SEQUENCE {
telco-id TelcoID, --- идентификатор оператора связи или структурного подразделения
dict ObjectDescriptor, --- тип справочника, по которому имеется информация
                       --- (идентификатор запроса справочника) (Requested...)
count INTEGER (1 .. 4294967295), --- количество записей в справочнике
change-dates FindRange --- минимальное и максимальное дата/время изменения записей в справочнике
}

--- отчет по наличию информации по местоположению абонентов
locationPresence TAGGED ::= {
  OID { sorm-report-presense-locations }
  DATASEQUENCEOFStandardInterval
}

--- интервал времени, на котором имеются данные по абонентам, соединениям и платежам, в рамках стандарта связи
StandardInterval ::= SEQUENCE {
telco-id TelcoID, --- идентификатор оператора связи или структурного подразделения
standard NetworkType, --- стандарт связи, информация по которому имеется в ИС
range FindRange, --- интервал времени, на который имеются данные в ИС
count INTEGER OPTIONAL --- количество записей
```

```
}

END
```

16. RequestedAbonents.asn

```
RequestedAbonents DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS RequestedAbonent;

IMPORTS TAGGED,
        sorm-request-abonent-person,
        sorm-request-abonent-organization
        FROM Classification

        RequestedAddress FROM Addresses;

RequestedAbonent ::= SEQUENCE {
 id   TAGGED.&id ({RequestedAbonentVariants}),
 data TAGGED.&Data ({RequestedAbonentVariants}{@id})
}

--- варианты запрашиваемых идентификаторов
RequestedAbonentVariantsTAGGED ::= {
requestedPerson |                                   --- физическое лицо
requestedOrganization                               --- юридическое лицо
}

--- поля параметра запроса на физическое лицо
requestedPerson TAGGED ::= {
 OID { sorm-request-abonent-person }
 DATA RequestedPerson
}

RequestedPerson ::= SEQUENCE {
 given-name [0]    UTF8String (SIZE (1 .. 256)) OPTIONAL,  --- имя
 initial [1]   UTF8String (SIZE (1 .. 256)) OPTIONAL,      --- отчество
(при наличии)
 family-name [2]  UTF8String (SIZE (1 .. 256)) OPTIONAL,  --- фамилия
 passport-info [3] RequestedPassport OPTIONAL,            --- паспортные
ланные
 address [4] RequestedAddress OPTIONAL,                   --- адресные
данные: адрес регистрации по месту жительства (пребывания), адрес установки
пользовательского устройства (телефонного аппарата)
 contract [6]      UTF8String (SIZE (1 .. 64))    OPTIONAL,---      номер
договора
```

```
        birth-date [7]          GeneralizedTime OPTIONAL,           ---          дата
рождения
        e-mail [8]          UTF8String (SIZE (1 .. 256)) OPTIONAL, ---          адрес
электронной почты
        phone-contact [9] UTF8String (SIZE (1 .. 128)) OPTIONAL, ---     телефоны
для взаимодействия абонента с оператором связи при оказании услуг связи (при
наличии)
        ssid [10]           UTF8String (SIZE (1 .. 256)) OPTIONAL, ---          SSID
абонентской WiFI-точки доступа
        mac [11]            OCTET STRING (SIZE (6)) OPTIONAL          --- MAC-
адрес абонентской WiFI-точки доступа
        }
        --- поля паспортных данных
        RequestedPassport ::= SEQUENCE {
         doc-type-id [0]            INTEGER (0 .. 65535) OPTIONAL,          ---
идентификатор типа документа, удостоверяющего личность
         passport-serial [1]        UTF8String (SIZE (1..16)) OPTIONAL,    --- серия
паспорта
         passport-number [2]    UTF8String (SIZE (1..16)) OPTIONAL     --- номер
паспорта
        }

        -- поля параметра запроса на юридическое лицо
        requestedOrganization TAGGED ::= {
         OID { sorm-request-abonent-organization }
         DATA RequestedOrganization
        }

        RequestedOrganization ::= SEQUENCE {
         full-name [0]      UTF8String (SIZE (1 .. 256)) OPTIONAL,          ---
полное наименование организации
         address [1] RequestedAddressOPTIONAL,      ---      адресные данные:
адрес юридического лица в пределах места нахождения; адрес доставки счета;
адрес установки пользовательского устройства (телефонного аппарата)
         inn [2]            NumericString (SIZE (1 .. 64)) OPTIONAL,       --- ИНН
         internal-user [3]  UTF8String (SIZE (1 .. 64)) OPTIONAL,          ---
внутренний пользователь
         contract [4]       UTF8String (SIZE (1 .. 64)) OPTIONAL,          --- номер
договора
         e-mail [5]         UTF8String (SIZE (1 .. 256)) OPTIONAL, ---          адрес
электронной почты
         phone-fax [6]      UTF8String (SIZE (1 .. 128)) OPTIONAL, ---     телефоны,
факс (при наличии) для взаимодействия абонента с оператором связи по
вопросам оказания услуг связи
         ssid [7]           UTF8String (SIZE (1 .. 256)) OPTIONAL,          ---
SSID абонентской WiFI-точки доступа
```

```
  mac [8]          OCTET STRING (SIZE (6)) OPTIONAL
--- MAC-адрес абонентской WiFI-точки доступа
 }

 END
```

17. RequestedConnections.asn

```
RequestedConnections DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS RequestedConnection;

IMPORTS TAGGED,
        sorm-request-connection-pstn,
        sorm-request-connection-mobile,
        sorm-request-connection-entrance,
        sorm-request-connection-aaa-login,
        sorm-request-connection-resource,
        sorm-request-connection-email,
        sorm-request-connection-im,
        sorm-request-connection-voip,
        sorm-request-connection-file-transfer,
        sorm-request-connection-term-access,
        sorm-request-connection-raw-flows,
        sorm-request-connection-address-translations,
        sorm-request-connection-sms
        FROM Classification

        requestedPagerIdentifier,
        requestedPstnIdentifier,
        requestedGsmIdentifier,
        requestedCdmaIdentifier,
        requestedDataNetworkIdentifier
        FROM RequestedIdentifiers

        PhoneAbonentType
        FROM Dictionaries

        Bunch,
        DataNetworkEquipment,
        IPAddress,
        NetworkPeerInfo,
        DataVoipNumber,
        VoipProtocol,
        IMProtocol,
        HttpMethod,
```

```
        EntranceEventType
        FROM NetworkIdentifiers

        Location
        FROM Locations;

    RequestedConnection ::= SEQUENCE {
     id   TAGGED.&id ({RequestedConnectionVariants}),
     data TAGGED.&Data ({RequestedConnectionVariants}{@id})
    }

    --- варианты запрашиваемых параметров связей
    RequestedConnectionVariants TAGGED ::= {
     requestedPagerIdentifier
     | requestedConnectionPstn                    --- параметры соединений
абонента телефонной сети связи
     | requestedConnectionMobile          --- параметры соединений
абонента сети подвижной радиотелефонной связи
     | requestedConnectionEntrance                --- параметры соединений,
связанных с действиями в личном кабинете
     | requestedAAALogin
     | requestedResource
     | requestedEmail
     | requestedIm
     | requestedVoip
     | requestedFileTransfer
     | requestedTermAccess
     | requestedRawFlows
     | requestedAddressTranslations
     | requestedConnectionSms
    }

    --- параметры соединений, связанных с действиями в личном кабинете
    requestedConnectionEntrance TAGGED ::= {
     OID { sorm-request-connection-entrance }
     DATACHOICE {
    protocol-code [0]  INTEGER (0 .. 65535),      --- код протокола в
соответствии с RFC1700
    client-info [1]    NetworkPeerInfo,           --- идентификатор
абонента сети передачи данных
    server-info [2]    NetworkPeerInfo,           --- идентификатор
сервера сети передачи данных
    abonent-id [3]          RequestedConnectionEntranceIdentifier,      ---
идентификаторы вызывающего абонента
    owner-id [4]       RequestedConnectionEntranceIdentifier,      ---
идентификаторы вызываемого абонента
```

```
        event-type [5]  EntranceEventType,                  --- тип события
        user-agent [6] UTF8String (SIZE (1 .. 8192)),   --- заголовок User-agent HTTP
соединения
        service-id [7] INTEGER (0 .. 4294967295), --- идентификатор заказанной
услуги
        service-ip-address [8] IPAddress, --- внешний IP-адрес для подключенной
услуги
        service-domain [9] UTF8String (SIZE (2 .. 256)), --- название
подключенного/отключенного/удаленного домена
        service-e-mail [10] UTF8String (SIZE (1 .. 256)), --- адрес подключенной
электронной почты
        service-parameter [11]  UTF8String (SIZE (1 .. 8192)), --- индивидуальные
параметры настройки услуги абонента
        location [12]       Location                        --- местоположение
абонента
         }
        }

        -- Идентификаторы Entrance
        RequestedConnectionEntranceIdentifier ::= SEQUENCE {
         id  TAGGED.&id ({RequestedConnectionEntranceIdentifierVariants}),
         data TAGGED.&Data
({RequestedConnectionEntranceIdentifierVariants}{@id})
        }

        RequestedConnectionEntranceIdentifierVariants TAGGED ::= {
         requestedDataNetworkIdentifier
        }

        --- параметры соединений абонента телефонной сети связи
        requestedConnectionPstn TAGGED ::= {
         OID { sorm-request-connection-pstn }
         DATA CHOICE {
        duration [0]        INTEGER (0 .. 86399),           --- время соединения
        call-type-id [1]        INTEGER  (0 .. 4294967295),     ---     тип
соединения
        in-abonent-type [2]     PhoneAbonentType,               ---     тип
вызывающего абонента
        out-abonent-type [3]    PhoneAbonentType,               ---     тип
вызываемого абонента
        switch-id [4]           UTF8String (SIZE (1 .. 128)),   ---     код
коммутатора, обслужившего вызов
        inbound-bunch [5] UTF8String (SIZE (1 .. 32)),      --- входящий пучок
        outbound-bunch [6]      UTF8String (SIZE (1 .. 32)),    ---  исходящий
пучок
```

```
        border-switch-id [7]      UTF8String (SIZE (1 .. 128)),        ---        код
пограничного коммутатора
        term-cause [8]            INTEGER (0 .. 16384),                ---      причина
завершения соединения
        supplement-service-id [9]      INTEGER (0 .. 4294967295),  ---  ДВО  при
соединении
        phone-card-number [10] NumericString (SIZE (1.. 20)),       ---        номер
телефонной карты
        in-info [11]          RequestedConnectionPstnIdentifier,     ---
идентификаторы вызывающего абонента
        out-info [12]             RequestedConnectionPstnIdentifier,  ---
идентификаторы вызываемого абонента
        forwarding-identifier [13] UTF8String (SIZE (2 .. 32)),      --- телефонный
номер при переадресации
        dialed-digits [21]  UTF8String (SIZE (1 .. 128)),        --- набранный номер
вызываемого абонента
        message [20]              UTF8String                         ---        текстовое
содержание сообщения абонента
        }
        }

        -- Идентификаторы PSTN
        RequestedConnectionPstnIdentifier ::= SEQUENCE {
         id   TAGGED.&id ({RequestedConnectionPstnIdentifierVariants}),
         data TAGGED.&Data ({RequestedConnectionPstnIdentifierVariants}{@id})
        }

        RequestedConnectionPstnIdentifierVariants TAGGED ::= {
         requestedPstnIdentifier
        }

        requestedConnectionMobile TAGGED ::= {
         OID { sorm-request-connection-mobile }
         DATA CHOICE {
        duration [0]          INTEGER (0 .. 86399),                --- время соединения
        call-type-id [1]          INTEGER (0 .. 4294967295),  --- тип соединения
        supplement-service-id [2] INTEGER (0 .. 4294967295), ---      ДВО      при
соединении
        in-abonent-type [3]       PhoneAbonentType,              --- тип вызывающего
абонента
        out-abonent-type [4]      PhoneAbonentType,              --- тип вызываемого
абонента
        switch-id [5]             UTF8String (SIZE (1 .. 128)), --- код коммутатора,
обслужившего соединение
        --- или номер SMS центра, если SMS
        inbound-bunch [6]Bunch,                                  --- входящий пучок
```

```
        outbound-bunch [7]      Bunch,                          --- исходящий пучок
        border-switch-id [8]    UTF8String (SIZE (1 .. 128)),   --- код пограничного
коммутатора
        roaming-partner-id [9]  INTEGER (0 .. 4294967295),      --- код роумингового
партнера
        term-cause [10]         INTEGER (0 .. 16384),           ---          причина
завершения соединения
        in-info [11] RequestedConnectionMobileIdentifier,       --- идентификаторы
вызывающего абонента
        in-end-location [12]    Location,                       --- местоположение
вызывающего абонента при завершении вызова
        in-begin-location [13]  Location,                       --- местоположение
вызывающего абонента на начало вызова
        dialed-digits [14]  UTF8String (SIZE (1 .. 128)),       --- набранный номер
вызываемого абонента
        out-info [15] RequestedConnectionMobileIdentifier,      --- идентификаторы
вызываемого абонента
        out-begin-location [16] Location,                       --- местоположение
вызываемого абонента на начало вызова
        out-end-location [17]   Location,                       --- местоположение
вызываемого абонента при завершении вызова
        forwarding-identifier [18] UTF8String (SIZE (2 .. 32))  ---          телефонный
номер при переадресации
         }
        }

        requestedConnectionSms TAGGED ::= {
         OID { sorm-request-connection-sms }
         DATA CHOICE {
        call-type-id [1]        INTEGER (0 .. 4294967295),  --- тип соединения
        in-abonent-type [3]     PhoneAbonentType,           --- тип вызывающего
абонента
        out-abonent-type [4]    PhoneAbonentType,           --- тип вызываемого
абонента
        switch-id [5]           UTF8String (SIZE (1 .. 128)),  --- код коммутатора,
обслужившего соединение или номер SMS центра, если SMS
        inbound-bunch [6] Bunch,                            --- входящий пучок
        outbound-bunch [7]      Bunch,                      --- исходящий пучок
        border-switch-id [8]    UTF8String (SIZE (1 .. 128)),  --- код пограничного
коммутатора
        roaming-partner-id [9]  INTEGER (0 .. 4294967295),  --- код роумингового
партнера
        term-cause [10]         INTEGER (0 .. 16384),       ---          причина
завершения соединения
        in-info [11]        RequestedConnectionMobileIdentifier,    ---
идентификаторы вызывающего абонента
```

```
        in-end-location [12]    Location,                       --- местоположение
вызывающего абонента при завершении вызова
        in-begin-location [13]  Location,                       --- местоположение
вызывающего абонента на начало вызова
        out-info [15]       RequestedConnectionMobileIdentifier,    ---
идентификаторы вызываемого абонента
        out-begin-location [16] Location,               ---     местоположение
вызываемого абонента на начало вызова
        out-end-location [17]   Location,                       --- местоположение
вызываемого абонента при завершении вызова
        message [40]            UTF8String          --- текстовое содержание
сообщения абонента
          }
        }

        -- Идентификаторы мобильных абонентов
        RequestedConnectionMobileIdentifier ::= SEQUENCE {
        id    TAGGED.&id ({RequestedConnectionMobileIdentifierVariants}),
        data TAGGED.&Data
({RequestedConnectionMobileIdentifierVariants}{@id})
        }

        RequestedConnectionMobileIdentifierVariants TAGGED ::= {
         requestedGsmIdentifier |
         requestedCdmaIdentifier
        }

        requestedAAALogin TAGGED ::= {
         OID { sorm-request-connection-aaa-login }
         DATA CHOICE {
        point-id [0]  INTEGER (0 .. 1000),              --- идентификатор точки
подключения к сети передачи данных, с которой получены записи
        login-type [1]      ENUMERATED {                --- тип соединения
        connect (0),                                    --- подключение к сети передачи
данных
        disconnect (1),                         --- отключение от сети передачи
данных
        update (2)
          },
        user-equipment [2]      DataNetworkEquipment,       --- идентификатор
пользовательского оборудования
        allocated-ip [3]        IPAddress,          ---         выделенный
динамический IP-адрес
        user-name [4]           UTF8String (SIZE (1 .. 128)),  --- имя пользователя
(login)
```

user-password [5] UTF8String (SIZE (1 .. 128)), --- пользовательский пароль
connect-type [6] INTEGER (1 .. 65535), --- код протокола в соответствии с RFC1700 либо номер порта для TCP/UDP
calling-number [7]UTF8String (SIZE (2 .. 32)), --- вызывающий номер
called-number [8] UTF8String (SIZE (2 .. 32)), --- вызываемый номер
nas [9] NetworkPeerInfo, --- IP-адрес/порт NAS-сервера
apn [10] UTF8String (SIZE (1 .. 128)), --- наименование точки доступа (Access Point Name)
sgsn-ip [11] IPAddress, --- IP-адрес GPRS/EDGE SGSN
ggsn-ip [12] IPAddress, --- IP-адрес GPRS/EDGE GGSN
service-area-code [13] INTEGER (0 .. 65535), --- код зоны обслуживания (SAC) GPRS/EDGE
location-start [14] Location, --- базовая станция, посредством которой установлено соединение (передача данных в сети подвижной радиотелефонной связи)
location-end [15] Location, --- базовая станция, посредством которой завершено соединение (передача данных в сети подвижной радиотелефонной связи)
phone-card-number [16] NumericString (SIZE (20)), --- номер телефонной карты
imsi [17] NumericString (SIZE (2 .. 18)), --- IMSI мобильного абонента
imei [18] NumericString (SIZE (2 .. 18)), --- идентификатор мобильной станции абонента
esn [19] NumericString (SIZE (2 .. 18)), --- идентификатор мобильной станции абонента
pool-number [20] UTF8String(SIZE (1 .. 64)), --- номер модемного пула
mcc [21] UTF8String,
mnc [22] UTF8String
}
}

requestedResource TAGGED ::= {
OID { sorm-request-connection-resource }
DATA CHOICE {
point-id [0] INTEGER (0 .. 1000), --- идентификатор точки подключения к сети передачи данных, с которой получены записи
client-info [1] NetworkPeerInfo, --- идентификатор абонента сети передачи данных
server-info [2] NetworkPeerInfo, --- идентификатор сервера сети передачи данных
url [3] UTF8String (SIZE (1 .. 8192)), --- наименование информационного ресурса

```
        term-cause [4]     INTEGER (0 .. 16384),          --- причина
завершения соединения
        http-method [5]    HttpMethod,
        abonent-id [6]     UTF8String (SIZE (0 .. 64)),
        nat-info [10]      NetworkPeerInfo,               --- транслированные
NAT IP/порт
        location [11]      Location                       --- местоположение
абонента
         }
        }

        requestedEmail TAGGED ::= {
         OID { sorm-request-connection-email }
         DATA CHOICE {
        point-id [0] INTEGER (0 .. 1000),          --- идентификатор точки
подключения к сети передачи данных, с которой получены записи
        client-info [1]    NetworkPeerInfo,            --- идентификатор
абонента сети передачи данных
        server-info [2]    NetworkPeerInfo,            --- идентификатор сервера
сети передачи данных
        sender [3]   UTF8String (SIZE (1 .. 512)),     --- отправитель почтового
сообщения
        receiver [4] UTF8String (SIZE (1 .. 512)),     --- получатель почтового
сообщения
        cc [5]       UTF8String (SIZE (1 .. 512)),     --- получатель-копия
почтового сообщения
        subject [6]  UTF8String (SIZE (1 .. 2048)),    --- тема почтового
сообщения
        attachements [7]   BOOLEAN,                    --- наличие прикрепленных
файлов в письме (да/нет)
        mail-server [8]    UTF8String (SIZE (1 .. 512)), --- наименование сервера,
посредством которого отправлено письмо (в том числе сервер веб-почты)
        term-cause [9]     INTEGER (0 .. 16384),       --- причина завершения
соединения
        abonent-id [10]    UTF8String (SIZE (0 .. 64)),
        message [20]       UTF8String,                 --- текстовое содержание
сообщения абонента
        nat-info [21]      NetworkPeerInfo,            --- транслированные NAT
IP/порт
        location [22]      Location                    --- местоположение
абонента
         }
        }

        requestedIm TAGGED ::= {
         OID { sorm-request-connection-im }
```

```
  DATA CHOICE {
  point-id [0]          INTEGER (0 .. 1000),          --- идентификатор
точки подключения к сети передачи данных, с которой получены записи
  client-info [1]       NetworkPeerInfo,              --- идентификатор
абонента сети передачи данных
  server-info [2]       NetworkPeerInfo,              --- идентификатор
сервера сети передачи данных
  user-login [3]        UTF8String (SIZE (1 .. 128)), --- учетная запись
пользователя при подключении
  user-password [4]     UTF8String (SIZE (1 .. 128)), --- пользовательский
пароль при подключении
  sender-screen-name [5] UTF8String (SIZE (1 .. 256)), --- общедоступное
имя отправителя
  sender-uin [6]        UTF8String (SIZE (1 .. 256)), --- пользовательский
идентификатор отправителя (в том числе для сервера, предназначенного для
отправки мгновенных сообщений)
  receiver-screen-name [7] UTF8String (SIZE (1 .. 256)), --- общедоступное
имя получателя
  receiver-uin [8]      UTF8String (SIZE (1 .. 256)), --- пользовательский
идентификатор получателя (в том числе для сервера, предназначенного для
отправки мгновенных сообщений)
  protocol [9]      IMProtocol,
  term-cause [10]       INTEGER (0 .. 16384),         --- причина
завершения соединения
  abonent-id [11]       UTF8String (SIZE (0 .. 64)),
  message [20]          UTF8String,                   --- текстовое
содержание сообщения абонента
  nat-info [21]         NetworkPeerInfo,              --- транслированные
NATIP/порт
  location [22]         Location                      --- местоположение
абонента
   }
  }

  requestedVoip TAGGED ::= {
   OID { sorm-request-connection-voip }
   DATA CHOICE {
  point-id [0]          INTEGER (0 .. 1000),          --- идентификатор
точки подключения к сети передачи данных, с которой получены записи
  client-info [1]       NetworkPeerInfo,              --- идентификатор
абонента сети передачи данных
  server-info [2]       NetworkPeerInfo,              --- идентификатор
сервера сети передачи данных
  duration [3]      INTEGER (0 .. 864000),        --- длительность разговора
(секунд)
```

```
        originator-name [4]       UTF8String (SIZE (1 .. 512)), ---     общедоступное
имя инициатора связи
        call-type-id [5]          INTEGER (0 .. 4294967295),    ---          способ
подключения
        voip-calling-number [6]   DataVoipNumber,               ---          номер
вызывающего абонента
        voip-called-number [7]    DataVoipNumber,               ---          номер
вызываемого абонента
        inbound-bunch [8] Bunch,                                --- входящий пучок
        outbound-bunch [9]        Bunch,                        --- исходящий пучок
        conference-id [10] UTF8String (SIZE (1..64)),    ---        идентификатор
конференции
        protocol [11]             VoipProtocol,
        term-cause [12]           INTEGER (0 .. 16384),         ---          причина
завершения соединения
        abonent-id [13]           UTF8String (SIZE (0 .. 64)),
        nat-info [20]             NetworkPeerInfo,              --- транслированные
NAT IP/порт
        location [21]             Location                      ---  местоположение
абонента
          }
        }

        requestedFileTransfer TAGGED ::= {
          OID { sorm-request-connection-file-transfer }
          DATA CHOICE {
        point-id [0]          INTEGER (0 .. 1000),              ---     идентификатор
точки подключения к сети передачи данных, с которой получены записи
        client-info [1]           NetworkPeerInfo,              ---     идентификатор
абонента сети передачи данных
        server-info [2]           NetworkPeerInfo,              ---     идентификатор
сервера сети передачи данных
        server-name [3]           UTF8String (SIZE (1 .. 256)), --- название сервера
        user-name [4]             UTF8String (SIZE (1 .. 128)), --- имя пользователя,
наименование учетной записи
        user-password [5]  UTF8String (SIZE (1 .. 256)), ---       пользовательский
пароль
        term-cause [6]            INTEGER (1 .. 16384),         ---          причина
завершения соединения
        abonent-id [7]            UTF8String (SIZE (0 .. 64)),
        nat-info [20]             NetworkPeerInfo,              --- транслированные
NAT IP/порт
        location [21]             Location                      ---  местоположение
абонента
          }
        }
```

```
requestedTermAccess TAGGED ::= {
 OID { sorm-request-connection-term-access }
 DATA CHOICE {
point-id [0]  INTEGER (0 .. 1000),                    --- идентификатор точки подключения к сети передачи данных, с которой получены записи
client-info [1]    NetworkPeerInfo,                   --- идентификатор абонента сети передачи данных
server-info [2]    NetworkPeerInfo,                   --- идентификатор сервера сети передачи данных
abonent-id [3]     UTF8String (SIZE (0 .. 64)),
nat-info [10]      NetworkPeerInfo,                   --- транслированные NAT IP/порт
location [11]      Location,                          --- местоположение абонента
sni [13]           UTF8String (SIZE (1 .. 128))       --- поле Server Name Indication/Common Name
}
}

--- идентификаторы для соединений передачи данных (закрытые протоколы обмена)
requestedRawFlows TAGGED::= {
 OID { sorm-request-connection-raw-flows }
 DATA CHOICE {
point-id [0]       INTEGER (0 .. 1000),               --- идентификатор точки подключения к сети передачи данных, с которой получены записи
protocol-code [1]  INTEGER (0 .. 65535),              --- код протокола в соответствии с RFC1700
client-info [2]          NetworkPeerInfo,             --- идентификатор абонента сети передачи данных
server-info [3]          NetworkPeerInfo,             --- идентификатор сервера сети передачи данных
abonent-id [4]           UTF8String (SIZE (0 .. 64)),
nat-info [10]            NetworkPeerInfo,             --- транслированные NAT IP/порт
location [11]            Location,                    --- местоположение абонента
sni [13]           UTF8String (SIZE (1 .. 128))   --- SNI/CN
}
}

--- идентификаторы для соединений передачи данных (закрытые протоколы обмена)
requestedAddressTranslations TAGGED::= {
 OID { sorm-request-connection-address-translations }
```

```
    DATA CHOICE {
    point-id [0]        INTEGER (0 .. 1000),   --- идентификатор точки подключения к сети передачи данных, с которой получена запись о соединении
    record-type [2]          ENUMERATED {--- тип записи о трансляции сетевого адреса
    session-start (0),                          --- начало сессии трансляции
    session-end (1)                             --- окончание сессии трансляции
     },
    private-ip [3]          NetworkPeerInfo, --- внутренний адрес
    public-ip [4]       NetworkPeerInfo,     --- внешний адрес
    destination-ip [5]  NetworkPeerInfo,     --- адрес назначения
    translation-type [6]     ENUMERATED {--- тип трансляции сетевых адресов
        static-nat (0),                      --- статическая
        dynamic-nat (1),                     --- динамическая
        source-nat (2),                      --- источника
        destination-nat (3),                 --- получателя
        pat (4)                              --- адрес-порт
      }
    }
  }

  END
```

18. RequestedIdentifiers.asn

```
RequestedIdentifiers DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS
  RequestedIdentifier,
  requestedPagerIdentifier,
  requestedPstnIdentifier,
  requestedGsmIdentifier,
  requestedCdmaIdentifier,
  requestedDataNetworkIdentifier
  ;

IMPORTS TAGGED,
    sorm-request-identifier-pager,
    sorm-request-identifier-pstn,
    sorm-request-identifier-gsm,
    sorm-request-identifier-cdma,
    sorm-request-identifier-data-network,
    sorm-request-identifier-voip
  FROM Classification
```

```
        DataNetworkEquipment,
        IPAddress
      FROM NetworkIdentifiers
      ;

    RequestedIdentifier ::= SEQUENCE {
     id   TAGGED.&id ({RequestedIdentifierVariants}),
     data TAGGED.&Data ({RequestedIdentifierVariants}{@id})
    }

    --- варианты запрашиваемых идентификаторов
    RequestedIdentifierVariants TAGGED ::= {
     requestedPagerIdentifier |                    --- идентификатор сети
персонального радиовызова
     requestedPstnIdentifier |                     --- идентификатор
телефонной сети связи
     requestedGsmIdentifier |                      --- идентификатор GSM
     requestedCdmaIdentifier |                     --- идентификатор CDMA
     requestedDataNetworkIdentifier |              --- идентификатор сети
передачи данных
     requestedVoipIdentifier                       --- идентификатор voip
    }

    --- идентификатор сети персонального радиовызова
    requestedPagerIdentifier TAGGED ::= {
     OID { sorm-request-identifier-pager }
     DATA RequestedPagerIdentifier
    }

    RequestedPagerIdentifier ::= NumericString (SIZE (2 .. 18))

    --- идентификатор телефонной сети общего пользования
    requestedPstnIdentifierTAGGED ::= {
    OID { sorm-request-identifier-pstn }
     DATA RequestedPstnIdentifier
    }

    RequestedPstnIdentifier ::= SEQUENCE {
     directory-numberUTF8String (SIZE (1 .. 32)),          --- телефонный
номер в международном формате
     internal-number    NumericString (SIZE (1 .. 32)) OPTIONAL---
дополнительный внутренний номер (при наличии)
    }

    -- идентификатор абонента GSM
    requestedGsmIdentifier TAGGED ::= {
```

```
        OID { sorm-request-identifier-gsm }
        DATA RequestedGsmIdentifier
      }

      RequestedGsmIdentifier ::= CHOICE {
        directory-number [0]   UTF8String (SIZE (1 .. 32)),  --- телефонный номер
в международном формате
      imsi [1]                 NumericString (SIZE (2 .. 18)),---   идентификатор
мобильного абонента
      imei [2]                 NumericString (SIZE (2 .. 18)),---   идентификатор
мобильной станции
      icc [3]            NumericString (SIZE (10 .. 20))      ---   идентификатор
SIM-карты абонента
      }

      -- идентификатор абонента CDMA
      requestedCdmaIdentifier TAGGED ::= {
        OID { sorm-request-identifier-cdma }
        DATA RequestedCdmaIdentifier
      }

      RequestedCdmaIdentifier ::= CHOICE {
        directory-number [0]   UTF8String (SIZE (1 .. 32)),  ---      телефонный
номер в международном формате
      imsi [1]                 NumericString (SIZE (2 .. 18)),---   идентификатор
мобильного абонента
      esn [2]                  NumericString (SIZE (2 .. 18)),---   идентификатор
мобильной станции
      min [3]                  NumericString (SIZE (2 .. 18)),---   идентификатор
мобильного абонента (CDMA)
      icc [4]            NumericString (SIZE (10 .. 20))      ---   идентификатор
SIM-карты абонента
      }

      -- Идентификатор сети передачи данных
      requestedDataNetworkIdentifier TAGGED ::= {
        OID { sorm-request-identifier-data-network }
        DATA RequestedDataNetworkIdentifier
      }

      RequestedDataNetworkIdentifier ::= CHOICE {
        user-equipment [0]     DataNetworkEquipment,      ---   идентификатор
пользовательского оборудования
        login [1]          UTF8String (SIZE (1 .. 128)),      --- имя пользователя
- login
        ip-address [2]         IPAddress,                 --- IP-адрес
```

```
        e-mail [3]          UTF8String (SIZE (1 .. 128)),          ---          адрес
электронной почты
        phone-number [5]    UTF8String (SIZE (2 .. 32)),    --- номер телефона
        user-domain [6]     UTF8String (SIZE (2 .. 256))    --- пользовательский
домен
    }

    -- Идентификатор voip
    requestedVoipIdentifier TAGGED ::= {
     OID { sorm-request-identifier-voip }
     DATA RequestedVoipIdentifier
    }

    RequestedVoipIdentifier ::= CHOICE {
     ip-address [0]          IPAddress,                    --- IP-адрес абонента
     originator-name [1]     UTF8String (SIZE (1 .. 512)), ---    общедоступное
имя инициатора связи
     calling-number [2]      UTF8String (SIZE (1 .. 32))   ---          номер
вызывающего абонента
    }

    END
```

19. Sessions.asn

```
    Sessions DEFINITIONS IMPLICIT TAGS ::=
    BEGIN

    EXPORTS sessionMessage;

    IMPORTS TAGGED
        ,sorm-message-session
        FROM Classification;

    sessionMessage TAGGED ::= {
     OID { sorm-message-session }
     DATA CHOICE {
      connect [0]           ConnectRequest,          ---      запрос      на
открытие сессии
      connect-response [1]  ConnectResponse,         --- ответ на запрос
открытия сессии

      adjustment [2]        AdjustmentRequest,       ---      согласование
поддерживаемых типов со стороны ПУ
      adjustment-response [3]    AdjustmentResponse, --- ответ на запрос
согласования данных
```

```
        disconnect [4]          DisconnectRequest,          --- запрос на
закрытие сессии
        disconnect-response [5]     DisconnectResponse      --- ответ на запрос
закрытия сессии
        }
    }

    --- запрос создания сессии
    ConnectRequest ::= SEQUENCE {
      session-timeout          INTEGER (60 .. 2592000),     --- максимальное
время неактивности
    max-data-length            INTEGER (10 .. 100000),      --- максимальная
длина блока отчета (в строках)
    data-packet-window-size    INTEGER (4 .. 256),          --- окно канала
передачи данных – максимальное число блоков данных, которое может быть
отправлено без подтверждения приема
    data-load-timeout          INTEGER (1 .. 60),           --- таймаут начала
передачи блоков отчетов
    request-response-timeout   INTEGER (1 .. 60),           --- таймаут ответа на
запрос
    data-packet-response-timeout  INTEGER (1 .. 60)         --- таймаут
подтверждения приема блока данных отчета
    }

    -- ответ на запрос создания сессии
    ConnectResponse ::= SEQUENCE {
    confirmed-data-packet-window-size  INTEGER (4 .. 256),  ---
подтвержденное окно передачи данных – окно, обеспечивающее
функционирование ИС ОРМ и должно быть меньше или равно окну,
переданному в ConnectRequest
    confirmed-session-timeout          INTEGER (60 .. 2592000),  ---
подтвержденное максимальное время неактивности должно быть больше или
равно значению, переданному в ConnectRequest
      confirmed-data-load-timeoutINTEGER (1 .. 60),     --- подтвержденный
таймаут начала передачи блоков отчетов – должен быть больше или равен
значению, переданному в ConnectRequest
      confirmed-request-response-timeout    INTEGER (1 .. 60),    ---
подтвержденный таймаут ответа на запрос должен быть больше или равен
значению, переданному в ConnectRequest
    supportsSEQUENCEOFObjectDescriptor  ---  весь  список
поддерживаемых СОРМ типов запросов, типов отчётов
    }

    --- согласование поддерживаемых типов со стороны ПУ
    AdjustmentRequest ::= SEQUENCE {
```

Supports SEQUENCE OF ObjectDescriptor --- список поддерживаемых ПУ типов запросов, типов отчётов. Данный список должен быть меньшим или равным списку в сообщении ConnectRequest
}

-- ответ на согласование списка поддерживаемых типов
AdjustmentResponse ::= NULL --- ответ на запрос согласования данных

--- запрос завершения сессии
DisconnectRequest ::= NULL

--- ответ на запрос завершения сессии
DisconnectResponse ::= NULL

END

20. Sorm.asn

```
Sorm DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS DateAndTime,
        FindRange,
        MessageID,
        Message;

IMPORTS TAGGED FROM Classification
        sessionMessage FROM Sessions
        trapMessage FROM Traps
        taskMessage FROM Tasks
        reportMessage FROM Reports
        managementMessage FROM Management
        unformattedMessage FROM Unformatted
        filterMessage FROM Filters;


Version ::= PrintableString
vers Version ::= "3.3.0"          --- текущая версия протокола

--- Оболочка сообщения COPM
Message ::= SEQUENCE {
version       Version OPTIONAL,                    --- версия протокола
   message-id      MessageID,                      --- номер запроса
   message-time    DateAndTime,                    --- время и дата запроса
   operator-name   PrintableString (SIZE (1 .. 128)) OPTIONAL,      ---
наименование оператора связи
```

```
        id      TAGGED.&id ({SormPDUs}),        --- идентификтор блока данных
        data TAGGED.&Data({SormPDUs}{@id})      --- данные блока данных
        }

        -- Блок данных сообщения
        SormPDUsTAGGED ::= {
        sessionMessage                          --- сообщения организации сессии
            | trapMessage                       --- сообщения сигналов
            | taskMessage                       --- сообщения работы с задачами
            | reportMessage                     --- сообщения работы с отчётами
            | managementMessage                 --- сообщения канала передачи мониторинга (КПМ)
            | unformattedMessage                --- сообщения канала передачи неформатированных данных (КПНФ)
            | filterMessage                     --- сообщения установки/снятия фильтров записываемого содержимого соединений сети передачи данных
        }

        --- Общие данные

        -- Номер сообщения
        MessageID ::= INTEGER (0 .. 4294967295)

        -- Дата и время
        DateAndTime ::= UTCTime

        -- Диапазон поиска
        FindRange ::= SEQUENCE {
          begin-find [0]    DateAndTime OPTIONAL,       --- время и дата начала поиска информации
          end-find [1]      DateAndTimeOPTIONAL         --- время и дата окончания поиска информации
        }

        END
```

21. TasksAbonents.asn

```
        TasksAbonents DEFINITIONS IMPLICIT TAGS ::=
        BEGIN

        EXPORTS AbonentsTask;

        IMPORTS
```

LogicalOperation FROM Tasks
RequestedIdentifier FROM RequestedIdentifiers
RequestedAbonent FROM RequestedAbonents;

AbonentsTask ::= CHOICE {
validate-abonents-task [0] ValidateAbonentsTask, --- задача на поиск информации о принадлежности идентификаторов абонентов сети оператора связи

validate-identifiers [1] ValidateIdentifiersTask, --- задача на поиск информации об идентификаторах абонентов сети оператора связи, зарегистрированных на физическое или юридическое лицо

validate-services [2] ValidateServicesTask --- задача на поиск информации о доступных абоненту видам услуг связи
}

--- задача на поиск информации о принадлежности идентификаторов абонентов сети оператора связи
ValidateAbonentsTask ::= RequestedIdentifiers --- запрашиваемые идентификаторы

RequestedIdentifiers ::= SEQUENCEOFRequestedIdentifierParameters -- последовательность из идентификаторов и логических операций

RequestedIdentifierParameters ::= CHOICE {
separator [0] LogicalOperation, --- логическая операция или скобка
find-mask [1] RequestedIdentifier --- параметр – идентификатор
}

--- задача на поиск информации об идентификаторах абонентов сети оператора связи зарегистрированных на физическое или юридическое лицо
ValidateIdentifiersTask ::= RequestedAbonents --- запрашиваемые абоненты

RequestedAbonents ::= SEQUENCEOFRequestedAbonentsParameters
--- последовательность логических операций и параметров

RequestedAbonentsParameters ::= CHOICE {
separator [0] LogicalOperation, --- логический оператор связки
find-mask [1] RequestedAbonent --- информация запроса об абоненте
}

```
--- задача на поиск истории услуг связи, оказанных абоненту
--- запрос по маске запрещён, идентификатор указывается полностью
ValidateServicesTask ::= SEQUENCE OF ValidateServicesParameters
ValidateServicesParameters ::= CHOICE {
 separator [0]    LogicalOperation,                --- логическая операция или скобка
 find-mask [1]    ValidateServicesParameter        --- параметр – идентификатор
}

ValidateServicesParameter ::= CHOICE {
 contract [0]     UTF8String (SIZE (1 .. 64)),     --- номер договора
 identifier [1]   RequestedIdentifier
}

END
```

22. Tasks.asn

```
Tasks DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS taskMessage,
      TaskID,
      LogicalOperation,
      CreateTaskResponse,
      DataContentID;

IMPORTS TAGGED,
      sorm-message-task
   FROM Classification

      FindRange,
      MessageID
   FROM Sorm

      TelcoList,
      DictionaryTask
   FROM Dictionaries

      AbonentsTask
   FROM TasksAbonents

      ConnectionsTask
   FROM TasksConnections

      LocationTask
```

```
        FROM TasksLocation

          PaymentsTask
        FROM TasksPayments

          PresenseTask
        FROM TasksPresense

          DataContentTask
        FROM TasksContentTask

          NonFormalizedTaskRequest,
          NonFormalizedTaskResponse
        FROM
          TasksNonFormalized;

      taskMessage TAGGED ::= {
       OID {sorm-message-task}
       DATA CHOICE {
         data-ready-request [0]          DataReadyRequest,          --- запрос готовности
данных
         data-ready-response [1]         DataReadyResponse,         ---          ответ
на запрос готовности данных
         data-load-request [2]           DataLoadRequest,           --- запрос загрузки
данных
         data-load-response [3]          DataLoadResponse,          --- ответ на запрос
загрузки данных
         data-drop-request [4]           DataDropRequest,           --- запрос удаления
данных
         data-drop-response [5]          DataDropResponse,          --- ответ на запрос
удаления данных
       data-interrupt-request [6]        DataInterruptRequest,      ---          запрос
прерывания загрузки данных
         data-interrupt-response [7]     DataInterruptResponse,     ---          ответ
на запрос прерывания загрузки данных
         create-task-request [8]         CreateTaskRequest,         ---          запрос
на создание задачи по обработке информации
         create-task-response [9]        CreateTaskResponse,        ---          ответ
на запрос создания задачи
         non-formalized-task-request [10]  NonFormalizedTaskRequest,  --- запрос
на создание задачи по обработке неформализованных данных
         non-formalized-task-response [11] NonFormalizedTaskResponse --- ответ
на запрос создания задачи по обработке неформализованных данных
        }
      }
```

```
--- в этом запросе не параметров
DataReadyRequest ::= NULL

--- запрос загрузки данных конкретной задачи
DataLoadRequest ::= TaskID

--- запрос удаления данных конкретной задачи
DataDropRequest ::= TaskID

--- запрос прерывания загрузки данных
DataInterruptRequest ::=  TaskID

--- запрос на создание задачи поиска
CreateTaskRequest ::= SEQUENCE {
telcos [0]    TelcoList OPTIONAL,                --- список операторов
связи
range [1]     FindRange OPTIONAL,                --- временной диапазон
поиска
report-limit [2]     INTEGER (1 .. 10000000) OPTIONAL,     ---
ограничение на максимальное количество возвращаемых записей
task [3] CHOICE {
    dictionary [0] DictionaryTask,  --- задачи пополнения справочников
(нормативно-справочная информация)
    abonents [1]  AbonentsTask,     --- задачи поисков по принадлежности
абонентов
    connections [2]     ConnectionsTask, ---        задачи        поисков
по соединениям абонентов
    location [3]  LocationTask,     ---     задача     получения     данных
местоположения абонентов
    payments [4]  PaymentsTask,     --- задачи поисков по совершенным
платежам
    presense [6]  PresenseTask,     --- задачи предоставления сведений
о наличии данных
    data-content [9]     DataContentTask ---        задачи        получения
содержимого потоков
  }
}

-----------------------------------------------------------------------

--- последовательность записей о готовности данных задач
DataReadyResponse ::=  SEQUENCE OF DataReadyTaskRecord

DataReadyTaskRecord ::=      SEQUENCE {
task-id      TaskID,                        --- идентификатор задачи
result TaskResult                           --- результат выполнения задачи
```

```
}

TaskResult ::= SEQUENCE {
result ENUMERATED {
    data-not-ready (0),                 --- данные не обработаны, задача
в процессе выполнения
    data-ready (1),                     --- данные в наличии, задача
выполнена
    data-not-found (2),                 --- данных отсутствуют, задача
выполнена
    error (3)                           --- в процессе выполнения задачи
произошла ошибка
    },
report-records-number [0]   INTEGER (0 .. 999999999999) OPTIONAL,
--- для выполненной задачи – количество записей в отчете
report-limit-exeeded [1] BOOLEANOPTIONAL,--- количество записей
превысило лимит, заданный при создании задачи
error-description [2]   UTF8String (SIZE (1 .. 256)) OPTIONAL ---
описание произошедшей ошибки (в случае выявления ошибки)
}

DataLoadResponse ::= SEQUENCE {
  task-id               TaskID,           --- идентификатор задачи,
сгенерировавшей данный отчет
    data-exists         BOOLEAN,          --- признак существования
результатов исполнения задачи (имеются данные или отсутствуют)
data-blocks-number      INTEGER (0 .. 999999999999) OPTIONAL,   ---
количество блоков в отчете
error-description       UTF8String (SIZE (1 .. 256)) OPTIONAL ---
описание ошибки (в случае выявления ошибки)
}

DataDropResponse ::= SEQUENCE {
task-id                 TaskID,     --- идентификатор задачи, данные
которой будут удалены
successful          BOOLEAN,        --- признак корректного выполнения
запроса
error-description       UTF8String (SIZE (1 .. 256)) OPTIONAL ---
описание ошибки (в случае выявления ошибки)
}

DataInterruptResponse ::= SEQUENCE {
request-id          MessageID,      --- идентификатор прерванного
запроса загрузки данных
successful          BOOLEAN, --- признак корректного выполнения запроса
```

```
        data-blocks-available     INTEGER (0 .. 999999999999) OPTIONAL,      ---
количество оставшихся непереданными блоков
        error-description         UTF8String (SIZE (1 .. 256)) OPTIONAL   ---
описание ошибки (в случае выявления ошибки)
        }

        CreateTaskResponse ::= SEQUENCE {
        task-id                   TaskID OPTIONAL,      --- идентификатор задачи
        successful   BOOLEAN,                           ---      признак      корректного
выполнения запроса
        error-description UTF8String (SIZE (1 .. 256)) OPTIONAL    ---     описание
ошибки (в случае выявления ошибки)
        }

        ------------------------------------------------------------------------------
        -- идентификатор задачи
        TaskID ::= INTEGER (0 .. 4294967295)

        -- идентификатор потока
        DataContentID ::= UTF8String (SIZE (1 .. 512))

        LogicalOperation ::= ENUMERATED {
          operation-open-bracket (0),          -- открывающая скобка - "("
          operation-close-bracket (1),         -- закрывающая скобка - ")"
          operation-or (2),                    -- логическое "или"
          operation-and (3),                   -- логическое "и"
          operation-not (4)                    -- логическое "не"
        }

        END
```

23. TasksConnections.asn

```
        TasksConnections DEFINITIONS IMPLICIT TAGS ::=
        BEGIN

        EXPORTS ConnectionsTask;

        IMPORTS
           LogicalOperation FROM Tasks
           RequestedConnection FROM RequestedConnections
           ;

        ConnectionsTask ::= CHOICE {
          validate-connections [0]       ValidateConnectionsTask,      ---      задача      на
поиск соединений между абонентами в сети телефонной связи
```

validate-data [1] ValidateDataTask, --- задача на поиск соединений между абонентами сети передачи данных
validate-entrance [2] ValidateEntranceTask --- задача на поиск информации о действиях абонента в личном кабинете
}

--- задача на поиск по соединениям абонентов
ValidateConnectionsTask ::= RequestedConnectionIdentifiers --- запрашиваемые идентификаторы указываются все, кроме RequestedDataNetworkIdentifier
ValidateDataTask ::= RequestedConnectionIdentifiers --- запрашиваемые идентификаторы указываются RequestedDataNetworkIdentifier
ValidateEntranceTask ::= RequestedConnectionIdentifiers --- задача на поиск информации о действиях абонента в личном кабинете

RequestedConnectionIdentifiers ::= SEQUENCE OF RequestedConnectionParameter
RequestedConnectionParameter ::= CHOICE {
separator [0] LogicalOperation, --- логический оператор связки
find-mask [1] RequestedConnection --- параметр запроса
}

END

24. TasksLocation.asn

TasksLocation DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS LocationTask;

IMPORTS
IPAddress
FROM NetworkIdentifiers;

--- задача получения данных местоположения абонентов
LocationTask ::= RequestedLocationIdentifier --- запрашиваемые идентификаторы для определения местоположения

RequestedLocationIdentifier ::=
CHOICE {
directory-number [0] UTF8String (SIZE (1 .. 32)), --- телефонный номер в международном формате
imsi [1] NumericString (SIZE (2 .. 18)),--- идентификатор мобильного абонента
ip-address [2] IPAddress, --- IP-адрес абонента

```
        imei [3]        NumericString (SIZE (2 .. 18)) ---    идентификатор
мобильной станции
        }

END
```

25. TasksNonFormalized.asn

```
TasksNonFormalized DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS
    EntityId,
    NonFormalizedTaskRequest,
    NonFormalizedTaskResponse,
    NonFormalizedEntityAttributeData;

IMPORTS  TelcoList
    FROM Dictionaries

    FindRange,
    MessageID,
    DateAndTime
    FROM Sorm

    Location
    FROM Locations

    LogicalOperation,
    CreateTaskResponse
    FROM Tasks
  ;

-- TaskID,
    -- LogicalOperation;

NonFormalizedTaskRequest ::= CHOICE {
get-entities [0]        GetEntities,                    ---  тип  сообщения
"запрос получения списка типов сущностей"
get-attributes [1]      GetEntityAtttibutes,            ---  тип  сообщения
"запрос получения списка атрибутов сущности"
validate-task [2]       ValidateNonFormalizedTask,      ---  тип  сообщения
"задача поиска неформализованных данных"
validate-presense [3]   NonFormalizedPresenseTask       ---  тип  сообщения
"задача предоставления сведений о наличии неформализованных данных"
}
```

```
NonFormalizedTaskResponse ::= CHOICE {
entities [0]          GetEntitiesResponse,                  --- ответ на запрос получения списка типов сущностей
entity-attributes [1]     GetEntityAtttibutesResponse,  --- ответ на запрос получения списка атрибутов сущности
validate-task [2]     ValidateNonFormalizedTaskResponse,    --- ответ на запрос задачи поиска неформализованных данных
validate-presense [3]     NonFormalizedPresenseTaskResponse     --- ответ на запрос задачи предоставления сведений о наличии неформализованных данных
}

--- тип сообщения "запрос получения списка типов сущностей"
GetEntities ::= NULL

--- тип сообщения "запрос получения списка атрибутов сущности"
GetEntityAtttibutes ::= EntityId

--- тип сообщения "задача поиск неформализованных данных"
ValidateNonFormalizedTask ::= SEQUENCE {
entity-id             EntityId,                         --- сущность для поиска по неформализованным данным
parameters NonFormalizedParameters,          --- критерии поиска по неформализованным данным
range          FindRangeOPTIONAL,            --- временной диапазон поиска
report-limit INTEGER (1 .. 10000000) OPTIONAL    --- ограничение на максимальное количество возвращаемых записей
}

--- тип сообщения "задача предоставления сведений о наличии неформализованных данных"
NonFormalizedPresenseTask ::= EntityId

NonFormalizedParameters ::= SEQUENCE OF NonFormalizedParameter

NonFormalizedParameter ::= CHOICE {
 separator [0]      LogicalOperation,                 --- логическая операция
 find-mask [1]      NonFormalizedEntityCondition      --- условие
}

NonFormalizedEntityCondition ::= SEQUENCE {
 attribute          NonFormalizedEntityAttribute,     --- атрибут сущности
 operation   MathOperation,                           --- операция
 attribute-value    NonFormalizedEntityAttributeData  --- значение атрибута
```

```
}

NonFormalizedEntityAttribute ::= SEQUENCE {
  attribute-name    UTF8String (SIZE (1 .. 256)),      --- текстовое
наименование атрибута сущности
  attribute-type    AttributeType                      --- тип данных
атрибута
}

NonFormalizedEntityAttributeData ::= CHOICE {
datetime [0] DateAndTime,                  --- дата и время (час,
минута, секунда) с погрешностью ± 1 секунда
integer [1]  INTEGER,                      --- целочисленный
string [2]   UTF8String,                   --- строковый
boolean [3]  BOOLEAN,                      --- булевый
float [4]    REAL,                         --- с плавающей запятой
location [5] Location,                     --- местоположение
empty [6]    NULL                          --- не заполняется (null)
}

--- математические операции сравнения
MathOperation ::= ENUMERATED {
equal (0),              --- равно
less (1),               --- меньше
greater (2),            --- больше
not-equal (3),          --- не равно
less-or-equal (4),      --- меньше или равно
greater-or-equal (5)    --- больше или равно
}

-------------------------------------------------------------------------------
GetEntitiesResponse ::= SEQUENCE OF NonFormalizedEntity

NonFormalizedEntity ::= SEQUENCE {
entity-id          EntityId,                --- уникальный
идентификатор сущности
entity-name UTF8String (SIZE (1 .. 256))    --- текстовое наименование
сущности
}

GetEntityAtttibutesResponse ::=    SEQUENCE {
entity-id          EntityId,    --- уникальный идентификатор сущности
entity-attributes  SEQUENCE OF NonFormalizedEntityAttribute   ---
атрибуты сущности
}
```

```
ValidateNonFormalizedTaskResponse ::= CreateTaskResponse

NonFormalizedPresenseTaskResponse ::= CreateTaskResponse

--------------------------------------------------------------------------------

--- типы данных атрибутов
AttributeType ::= ENUMERATED {
  date-time (0),              --- дата и время (час, минута, секунда) с
погрешностью ± 1 секунда
  integer (1),                --- целочисленный
  string (2),                 --- строковый
boolean (3),                  --- булевый
float (4),                    --- с плавающей запятой
location (5),                 --- местоположение
empty (6)                     --- пустое значение
}

--- Идентификатор сущности
EntityId ::= INTEGER (0 .. 4294967296)

END
```

26. TasksPayments.asn

```
TasksPayments DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS PaymentsTask;

IMPORTS LogicalOperation
  FROM Tasks

    RequestedConnection
  FROM RequestedConnections

    RequestedIdentifier
  FROM RequestedIdentifiers

    RequestedAddress
  FROM Addresses

    TAGGED,
    sorm-request-payment-bank-transaction,
    sorm-request-payment-express-pays,
    sorm-request-payment-terminal-pays,
    sorm-request-payment-service-center,
```

```
        sorm-request-payment-cross-account,
        sorm-request-payment-telephone-card,
        sorm-request-payment-balance-fillups,
        sorm-request-payment-bank-division-transfer,
        sorm-request-payment-bank-card-transfer,
        sorm-request-payment-bank-account-transfer
      FROM Classification;

PaymentsTask ::= SEQUENCE {
 id   TAGGED.&id ({RequestedPaymentsVariants}),
 data TAGGED.&Data ({RequestedPaymentsVariants}{@id})
}

--- варианты запрашиваемых параметров связей
RequestedPaymentsVariantsTAGGED ::= {
bankTransactionTask
 | expressCardTask
 | publicTerminalTask
 | serviceCenterTask
 | crossAccountTask
 | telephoneCardTask
 | balanceFillupTask
 | bankDivisionTransferTask
 | bankCardTransferTask
 | bankAccountTransferTask
}

--- задача на поиск пополнения баланса посредством банковского перевода
bankTransactionTask TAGGED ::= {
 OID { sorm-request-payment-bank-transaction }
 DATA RequestedBankTransactionPays
}

RequestedBankTransactionPays ::= SEQUENCE OF
RequestedBankTransactionPaysParameters
--- последовательность логических операций и параметров
RequestedBankTransactionPaysParameters ::= CHOICE {
separator [0]      LogicalOperation,                --- логический оператор
связки
bank-account [1]   UTF8String (SIZE (1 .. 64)),     --- номер банковского
счета, с которого совершен платеж
bank-name [2]      UTF8String (SIZE (1 .. 512)),    --- наименование банка,
со счета которого совершен перевод
identifier [3]     RequestedIdentifier              --- идентификатор
абонента
}
```

```
--- задача на поиск пополнения баланса с использованием карты экспресс-оплаты
expressCardTask TAGGED ::= {
  OID { sorm-request-payment-express-pays }
  DATA RequestedExpressPays
}

RequestedExpressPays ::= SEQUENCE OF RequestedExpressPaysParameters
--- последовательность логических операций и параметров
RequestedExpressPaysParameters ::= CHOICE {
  separator [0]      LogicalOperation,                --- логический оператор связки
  express-card [1]   NumericString (SIZE (2 .. 20)),--- номер карты экспресс-оплаты
  identifier [2]     RequestedIdentifier              --- идентификатор абонента
}

--- задача на поиск пополнения баланса с использованием терминалов моментальных платежей
publicTerminalTask TAGGED ::= {
  OID { sorm-request-payment-terminal-pays }
  DATA RequestedTerminalPays
}

RequestedTerminalPays ::= SEQUENCE OF RequestedTerminalPaysParameters --- последовательность логических операций и параметров
RequestedTerminalPaysParameters ::= CHOICE {
  separator [0]          LogicalOperation,                --- логический оператор связки
  terminal-id [1]        UTF8String (SIZE (1 .. 64)),     --- идентификатор терминала
  terminal-number [2]    NumericString (SIZE (2 .. 20)),  --- номер терминала
  terminal-address [3]   RequestedAddress,                --- адрес терминала
  identifier [4]         RequestedIdentifier              --- идентификатор абонента
}

--- задача на поиск пополнения баланса посредством центров обслуживания клиентов (ЦОК)
serviceCenterTask TAGGED ::= {
  OID { sorm-request-payment-service-center }
  DATA RequestedServiceCenterPays
```

```
}

RequestedServiceCenterPays ::= SEQUENCE OF
RequestedServiceCenterPaysParameters --- последовательность логических
операций и параметров
RequestedServiceCenterPaysParameters ::= CHOICE {
separator [0] LogicalOperation, --- логический оператор
связки
center-id [1]UTF8String (SIZE (1 .. 64)), --- идентификатор центра
обслуживания клиентов
center-address [2] RequestedAddress, --- адрес центра обслуживания
клиентов
identifier [3] RequestedIdentifier --- идентификатор абонента
}

--- задача на поиск пополнения баланса посредством снятия денег со счета
другого абонента
crossAccountTask TAGGED ::= {
  OID { sorm-request-payment-cross-account }
  DATA RequestedCrossAccountPays
}

RequestedCrossAccountPays ::= SEQUENCE OF
RequestedCrossAccountPaysParameters --- последовательность
логических операций и параметров
RequestedCrossAccountPaysParameters ::= CHOICE {
separator [0] LogicalOperation, --- логический оператор
связки
source-identifier [1] RequestedIdentifier, --- идентификатор
абонента, со счета которого переводятся средства
dest-identifier [2] RequestedIdentifier --- идентификатор
абонента, на счет которого переводятся средства
}

--- задача на поиск пополнения баланса с использованием телефонных карт
telephoneCardTask TAGGED ::= {
  OID { sorm-request-payment-telephone-card }
  DATA RequestedTelephoneCardPays
}

RequestedTelephoneCardPays ::= SEQUENCE OF
RequestedTelephoneCardPaysParameters --- последовательность логических
операций и параметров
RequestedTelephoneCardPaysParameters ::= CHOICE {
 separator [0] LogicalOperation, --- логический оператор
связки
```

```
    card-number [1] NumericString (SIZE (2 .. 20)),--- номер телефонной
карты
    identifier [2]  RequestedIdentifier  ---  идентификатор
абонента
    }

    --- общая задача на поиск пополнения баланса личного счета абонента
    balanceFillupTask TAGGED ::= {
      OID { sorm-request-payment-balance-fillups }
      DATA RequestedBalanceFillups          --- параметры запроса
    }

    RequestedBalanceFillups  ::=  SEQUENCE  OF
RequestedBalanceFillupsParameters --- последовательность логических операций
и параметров
    RequestedBalanceFillupsParameters ::= CHOICE {
     separator [0]   LogicalOperation,     --- логический оператор связки
     identifier [1]  RequestedIdentifier   --- идентификатор абонента
    }

    --- задача на поиск перевода средств со счета абонента для их снятия
в отделении банка
    bankDivisionTransferTask TAGGED ::= {
    OID { sorm-request-payment-bank-division-transfer }
    DATA RequestedBankDivisionTranferPays      --- параметры запроса
    }

    RequestedBankDivisionTranferPays  ::=  SEQUENCE  OF
RequestedTranferParameters  ---  последовательность  логических  операций
и параметров

    --- задача на поиск перевода средств со счета абонента на банковскую
карту
    bankCardTransferTask TAGGED ::= {
      OID { sorm-request-payment-bank-card-transfer }
      DATA RequestedBankCardTranferPays        --- параметры запроса
    }

    RequestedBankCardTranferPays  ::=  SEQUENCE  OF
RequestedTranferParameters --- последовательность логических операций
и параметров

    --- задача на поиск перевода средств со счета абонента на счет в банке
    bankAccountTransferTask  TAGGED ::= {
      OID { sorm-request-payment-bank-account-transfer }
      DATA RequestedBankAccountTranferPays     --- параметры запроса
```

```
}

RequestedBankAccountTranferPays ::= SEQUENCE OF
RequestedTranferParameters --- последовательность логических операций
и параметров
RequestedTranferParameters ::= CHOICE {
 separator [0] LogicalOperation, --- логический
оператор связки
 source-identifier [1] RequestedIdentifier --- идентификатор
абонента инициатора перевода средств
}

END
```

27. TasksPresense.asn

```
TasksPresense DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS PresenseTask;

IMPORTS
    TAGGED,
    sorm-request-presense
    FROM Classification;

PresenseTask ::= SEQUENCE {
 id TAGGED.&id ({PresenseListVariants}),
 data TAGGED.&Data ({PresenseListVariants}{@id})
}

PresenseListVariants TAGGED ::= { presenseInfo }

presenseInfo TAGGED ::= {
 OID { sorm-request-presense }
 DATA ENUMERATED {
subscribers (0), --- запрос наличия информации об абонентах
и их идентификаторах
connections (1), --- запрос наличия информации о соединениях
payments (2), --- запрос наличия имеющейся информации
о платежах
dictionaries (3), --- запрос наличия справочников
locations (4) --- запрос наличия информации
о местоположении абонентов
}
}
```

```
END
```

28. Traps.asn

```
Traps DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS trapMessage;

IMPORTS
   TAGGED,
   sorm-message-trap
   FROM Classification

   MessageID
   FROM Sorm
   ;

trapMessage TAGGED ::= {
   OID { sorm-message-trap }
   DATA CHOICE {
    trap [0]  Trap,                                  --- тип сообщения "сигнал"
  trap-ack [1] TrapAck                               --- тип сообщения
"подтверждение сигнала"
   }
}

-- Блок данных сообщения типа "сигнал"
Trap ::= SEQUENCE {
 trap-type          TrapType,                -- тип сообщения
 trap-message           UTF8String (SIZE (1 .. 256)) OPTIONAL, --
описание сообщения
 reference-message      MessageIDOPTIONAL        -- номер сообщения,
к которому относится данный сигнал

}

TrapType ::= ENUMERATED {
heartbeat (0),                          -- тестовый пакет
restart-software (1),                   --  перезагрузка  программного
обеспечения (далее – ПО)
unauthorized-access (2),                --  попытка  несанкционированного
доступа
critical-error (3),                     --  ошибка  ПО,  данные  утеряны,
дальнейшая работа невозможна
major-error (4),                        --  ошибка  ПО,  данные  утеряны,
дальнейшая работа возможна
```

```
        minor-error (5)                    -- ошибка ПО, данные не утеряны,
дальнейшая работа возможна
        }

        -- Блок данных сообщения типа "подтверждение сигнала"
        -- Номер сообщения TrapAck должен соответствовать номеру сообщения
Trap
        TrapAck ::= NULL

        END
```

29. Unformatted.asn

```
        Unformatted DEFINITIONS IMPLICIT TAGS ::=

        BEGIN

        EXPORTS unformattedMessage;

        IMPORTS TAGGED,
            sorm-message-unformatted
          FROM Classification

            TelcoList
          FROM Dictionaries

            Acknowledgement
          FROM Reports

            CallsRecords
          FROM ReportsConnections

            DateAndTime,
            MessageID
          FROM Sorm;

        unformattedMessage TAGGED ::= {
          OID { sorm-message-unformatted }
          DATA CHOICE {
            request [0]    RawRequest,
            response [1]   RawResponse,
            report [2]             RawReport,
            report-ack [3]         RawAcknowledgement
          }
        }

        RawRequest ::= SEQUENCE {
```

```
        telcos      TelcoList,          --- список операторов связи
        raw-task    RawRequestTask --- запрос получения неформатированных
данных
        }

        RawRequestTask ::= CHOICE {
        data-types-request [0]    DataTypesRequest,    ---    запрос    проверки
наличия вида неформатированных данных в ИС ОРМ
        data-start-request [1]    DataStartRequest,          --- запрос на начало
передачи неформатированных данных
        data-stop-request [2]     DataStopRequest,           ---    запрос    на
прекращение передачи всех неформатированных данных
        data-stop-typed-request [3]    DataStopTypedRequest ---    запрос    на
прекращение передачи неформатированных данных по типу данных
        }

        DataStopTypedRequest ::= RawDataType

        --- типы данных, передаваемых ИС ОРМ
        RawDataType ::= ENUMERATED {
        raw-cdr (1),        ---    "сырые"    (неформатированные)    CDR-файлы,
полученные с коммутационного оборудования
        raw-ipdr (2),       --- "сырые" (неформатированные) IPDR-файлы
        raw-location (10), --- файлы с указанием местоположения в сетях
подвижной радиотелефонной связи
        raw-passive (11)  --- файлы со статистической информацией, полученной
от ТС ОРМ 86
        }

        DataTypesRequest ::= RawDataType

        ---Команда на начало передачи данных любого типа передается от ПУ
независимо от процесса передачи данных других типов, полученных от ИСБД
        DataStartRequest ::=  SEQUENCE {
        time-from   DateAndTime,    --- начало временного периода в буфере,
от которого необходимо получить данные
        time-to     DateAndTime,    --- окончание временного периода в буфере,
от которого необходимо получить данные
        raw-type    RawDataType     --- тип неформатированных данных передачи
        }

        DataStopRequest ::= NULL

        RawResponse ::= CHOICE {
        data-types-response [0]  DataTypesResponse,          ---          ответ
на запрос проверки наличия неформатированных вида данных в ИС ОРМ
```

data-start-response [1] DataStartResponse, --- ответ на запрос начала передачи неформатированных данных
data-stop-response [2] DataStopResponse, --- ответ на запрос остановки передачи неформатированных данных
data-stop-typed-response [3] DataStopTypedResponse--- ответ на запрос остановки передачи неформатированных данных по типу данных
}

DataStopTypedResponse ::= BOOLEAN --- признак корректного выполнения команды

DataTypesResponse ::= SEQUENCE {
successful BOOLEAN,--- признак наличия в ИС ОРМ запрошенного вида неформатированных данных
selected-type RawDataType, --- выбранный вид данных для передачи
time-from DateAndTime, --- начало временного периода в буфере, начиная с которого накоплены данные
time-to DateAndTime --- завершение временного периода в буфере, по которому накоплены данные
}

DataStartResponse ::= BOOLEAN --- признак положительного результата выполнения команды
DataStopResponse ::= BOOLEAN --- признак положительного результата выполнения команды

---в поле stream-id передается имя файла
RawReport ::= SEQUENCE {
request-id MessageID, --- идентификатор запроса
stream-id UTF8String (SIZE (1 .. 256)), --- имя передаваемого файла
total-blocks-number INTEGER (0 .. 999999999999), --- общее количество блоков в отчете
block-number INTEGER (1 .. 1000000000000), --- порядковый номер текущего блока
report-block RawDataBlock --- блок данных отчета (соответствует одному передаваемому файлу)
}

---блоки отчетов разных типов данных передаются независимо друг от друга.
RawDataBlock ::= CHOICE {
raw-cdr [1] RawBytesBlock, --- файлы с биллингом в сети телефонной связи, полученным от коммутационного оборудования

raw-ipdr [2] RawBytesBlock, --- файлы с биллингом в сети передачи данных
raw-location [10] RawBytesBlock, --- файлы с информацией о местоположении абонентов в сети подвижной радиотелефонной связи
raw-passive [11] RawBytesBlock --- файл с информацией, полученной от ТС ОРМ 86
}

RawBytesBlock::= SEQUENCE OF RawBytes
RawBytes ::= OCTET STRING (SIZE (1 .. 4096))

RawAcknowledgement ::= Acknowledgement

END

30. ReportsDataContent.asn

```
ReportsDataContent DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS DataContentReport;

IMPORTS TAGGED,
    sorm-report-data-content-raw
    FROM Classification;

DataContentReport ::= SEQUENCE {
 id   TAGGED.&id ({ReportedDataContentVariants}),
 data TAGGED.&Data ({ReportedDataContentVariants}{@id})
}

ReportedDataContentVariants TAGGED ::= {
 reportDataContentRaw
}

reportDataContentRaw TAGGED ::= {
 OID { sorm-report-data-content-raw }
 DATA SEQUENCE OF RawRecordContent
}

RawRecordContent ::= SEQUENCE {
successful   BOOLEAN,                                   --- признак корректного формирования блока данных
    data [0] OCTETSTRING (SIZE (1 .. 1048576)) OPTIONAL,  --- содержимое блока (в случае, если блок успешно сформирован)
    error [1] UTF8String (SIZE (1 .. 4096)) OPTIONAL,     --- описание ошибки (в случае, если не удалось сформировать блок)
```

codec-info [2] UTF8String (SIZE (1 .. 4096)) OPTIONAL, --- описание типа содержимого в поле data
direction [3] DataContentRawDirection OPTIONAL, --- направление передачи
channel [4] INTEGER OPTIONAL --- канал
}

DataContentRawDirection ::= ENUMERATED {
client-server (0),
server-client (1)
}

END

31. TasksContentTask.asn

```
TasksContentTask DEFINITIONS IMPLICIT TAGS ::=
BEGIN
EXPORTS
  DataContentTask
  ;

IMPORTS
  DataContentID
  FROM Tasks;

DataContentTask ::= DataContentID

END».
```