Приказ Минцифры России от 28.08.2023 N 750 "О внесении изменений в Требования к оборудованию и программно-техническим средствам, используемым организатором распространения информации в сети Интернет в эксплуатируемых им информационных системах, для проведения уполномоченными государственными органами, осуществляющими оперативно-розыскную деятельность или обеспечение безопасности Российской Федерации, мероприятий в целях реализации возложенных на них задач, утвержденные приказом Министерства цифрового развития, связи и массовых коммуникаций Российской Федерации от 29.10.2018 N 571 (Зарегистрировано в Минюсте России 27.11.2023 N 76124)"

МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ

И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПРИКАЗ

от 28 августа 2023 г. N 750

О ВНЕСЕНИИ ИЗМЕНЕНИЙ

В ТРЕБОВАНИЯ К ОБОРУДОВАНИЮ И ПРОГРАММНО-ТЕХНИЧЕСКИМ

СРЕДСТВАМ, ИСПОЛЬЗУЕМЫМ ОРГАНИЗАТОРОМ РАСПРОСТРАНЕНИЯ

ИНФОРМАЦИИ В СЕТИ "ИНТЕРНЕТ" В ЭКСПЛУАТИРУЕМЫХ ИМ

ИНФОРМАЦИОННЫХ СИСТЕМАХ, ДЛЯ ПРОВЕДЕНИЯ УПОЛНОМОЧЕННЫМИ

ГОСУДАРСТВЕННЫМИ ОРГАНАМИ, ОСУЩЕСТВЛЯЮЩИМИ

ОПЕРАТИВНО-РОЗЫСКНУЮ ДЕЯТЕЛЬНОСТЬ ИЛИ ОБЕСПЕЧЕНИЕ

БЕЗОПАСНОСТИ РОССИЙСКОЙ ФЕДЕРАЦИИ, МЕРОПРИЯТИЙ В ЦЕЛЯХ

РЕАЛИЗАЦИИ ВОЗЛОЖЕННЫХ НА НИХ ЗАДАЧ, УТВЕРЖДЕННЫЕ ПРИКАЗОМ

МИНИСТЕРСТВА ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ

КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ ОТ 29.10.2018 N 571

В соответствии с частью 4 статьи 10.1 Федерального закона от 27 июля 2006 г. N 149-ФЗ "Об информации, информационных технологиях и о защите информации", пунктами 5 и 8 Правил взаимодействия организаторов распространения информации в информационно-телекоммуникационной сети "Интернет" с уполномоченными государственными органами, осуществляющими оперативно-разыскную деятельность или обеспечение безопасности Российской Федерации, утвержденных постановлением Правительства Российской Федерации от 31 июля 2014 г. N 743, приказываю:

Утвердить прилагаемые изменения, которые вносятся в Требования к оборудованию и программно-техническим средствам, используемым организатором распространения информации в сети "Интернет" в эксплуатируемых им информационных системах, для проведения уполномоченными государственными органами, осуществляющими оперативно-розыскную деятельность или обеспечение безопасности Российской Федерации, мероприятий в целях реализации возложенных на них задач, утвержденные приказом Министерства цифрового развития, связи и массовых коммуникаций Российской Федерации от 29.10.2018 N 571 (зарегистрирован Министерством юстиции Российской Федерации 27 декабря 2018 г., регистрационный N 53207).

Министр

М.И.ШАДАЕВ

Утверждены

приказом Министерства

цифрового развития, связи

и массовых коммуникаций

Российской Федерации

от 28.08.2023 г. N 750

ИЗМЕНЕНИЯ

В ТРЕБОВАНИЯ К ОБОРУДОВАНИЮ И ПРОГРАММНО-ТЕХНИЧЕСКИМ СРЕДСТВАМ, ИСПОЛЬЗУЕМЫМ ОРГАНИЗАТОРОМ РАСПРОСТРАНЕНИЯ ИНФОРМАЦИИ В СЕТИ "ИНТЕРНЕТ" В ЭКСПЛУАТИРУЕМЫХ ИМ ИНФОРМАЦИОННЫХ СИСТЕМАХ, ДЛЯ ПРОВЕДЕНИЯ УПОЛНОМОЧЕННЫМИ ГОСУДАРСТВЕННЫМИ ОРГАНАМИ, ОСУЩЕСТВЛЯЮЩИМИ ОПЕРАТИВНО-РОЗЫСКНУЮ ДЕЯТЕЛЬНОСТЬ ИЛИ ОБЕСПЕЧЕНИЕ БЕЗОПАСНОСТИ РОССИЙСКОЙ ФЕДЕРАЦИИ, МЕРОПРИЯТИЙ В ЦЕЛЯХ РЕАЛИЗАЦИИ ВОЗЛОЖЕННЫХ НА НИХ ЗАДАЧ, УТВЕРЖДЕННЫЕ ПРИКАЗОМ

МИНИСТЕРСТВА ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ

КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

ОТ 29.10.2018 N 571

1. Пункт 4 изложить в следующей редакции:

"4. Поиск, обработка и передача на пункт управления уполномоченного подразделения органа федеральной службы безопасности (далее - ПУ) следующих данных, хранимых в ИС ОРИ и (или) ПТС ОРИ при реализации ПТС ОРИ в соответствии с пунктом 2 настоящих требований, осуществляются с использованием ПТС ОРИ по запросу уполномоченного подразделения органа федеральной службы безопасности или в автоматическом режиме:

1) информации о зарегистрированных пользователях в соответствии с пунктами 13 и 15 Правил хранения организаторами распространения информации в информационно-телекоммуникационной сети "Интернет" информации о фактах приема, передачи, доставки и (или) обработки голосовой информации, письменного текста, изображений, звуков, видео- или иных электронных сообщений пользователей информационно-телекоммуникационной сети "Интернет" и информации об этих пользователях и предоставления ее уполномоченным государственным органам, осуществляющим оперативно-разыскную деятельность или обеспечение безопасности Российской Федерации, утвержденных постановлением Правительства Российской Федерации от 23 сентября 2020 г. N 1526 (далее - Правила хранения N 1526);

2) информации о фактах приема, передачи, доставки и (или) обработки голосовой информации, письменного текста, изображений, звуков, видео- или иных электронных сообщений пользователей информационно-телекоммуникационной сети "Интернет" (далее - пользователь) и информации об этих пользователях в соответствии с пунктами 3, 14 и 15 Правил хранения N 1526;

3) текстовых сообщений пользователей информационно-телекоммуникационной сети "Интернет", голосовой информации, изображений, звуков, видео-, иных электронных сообщений пользователей в соответствии с Правилами хранения организатором распространения информации в информационно-телекоммуникационной сети "Интернет" текстовых сообщений пользователей информационно-телекоммуникационной сети "Интернет", голосовой информации, изображений, звуков, видео-, иных электронных сообщений пользователей информационно-телекоммуникационной сети "Интернет", утвержденными постановлением Правительства Российской Федерации от 26 февраля 2022 г. N 256, действующим до 1 сентября 2028 г.;

4) информации, необходимой для декодирования принимаемых, передаваемых, доставляемых и (или) обрабатываемых электронных сообщений;

5) информации о событиях, зафиксированных в системных журналах в соответствии с подпунктом "т" пункта 14 настоящих требований.".

2. Пункт 16 дополнить абзацем следующего содержания:

"Срок хранения информации об объектах контроля, отобранной по критериям отбора, для доставки объектов на ПУ должен составлять не менее 12 часов с момента отбора.".

3. Приложение N 4 изложить в следующей редакции:

"Приложение N 4

к Требованиям к оборудованию

и программно-техническим средствам,

используемым организатором

распространения информации в сети

"Интернет" в эксплуатируемых

им информационных системах,

для проведения уполномоченными

государственными органами,

осуществляющими оперативно-розыскную

деятельность или обеспечение

безопасности Российской Федерации,

мероприятий в целях реализации

возложенных на них задач,

утвержденным приказом Министерства

цифрового развития, связи и массовых

коммуникаций Российской Федерации

от 29.10.2018 N 571

ОПИСАНИЕ ПРОТОКОЛА ВЗАИМОДЕЙСТВИЯ ПУ И ПТС ОРИ

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-report-uni,
```

```
 sorm-request-connection-ori-data-standart,
 sorm-request-connection-ori-data-extended,
 sorm-request-connection-ori-link,
 sorm-request-connection-ori-aaa,
 sorm-request-connection-ori-streams-content,
 sorm-request-connection-ori-cert,
 sorm-request-connection-ori-keys,

 sorm-report-connection-ori-data,
 sorm-report-connection-ori-link,
 sorm-report-connection-ori-aaa,
 sorm-report-connection-ori-streams-content,
 sorm-report-connection-ori-cert,
 sorm-report-connection-ori-keys,
 sorm-request-abonent-ori,
 sorm-report-abonent-ori,

 sorm-request-dictionaries,
 sorm-report-dictionary-telcos,
 sorm-report-dictionary-ori-services,
 sorm-report-dictionary-ori-events,
 sorm-report-dictionary-ori-resources,
 sorm-report-dictionary-ori-user-types,
 sorm-report-dictionary-ori-payment-services,

 sorm-request-presense,
 sorm-report-presense-dictionaries,
 sorm-report-presense-abonents-ori,
 sorm-report-presense-connections-ori;

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-report-uni OID::= "289"

--- Параметры соединений
sorm-request-connection-ori-data-standart OID::= "265"
sorm-request-connection-ori-data-extended OID::= "266"
sorm-request-connection-ori-link OID::= "267"
sorm-request-connection-ori-aaa OID::= "268"
sorm-request-connection-ori-streams-content OID::= "269"
sorm-request-connection-ori-cert OID::= "270"
sorm-request-connection-ori-keys OID::= "271"

sorm-report-connection-ori-data OID::= "10"
sorm-report-connection-ori-link OID::= "11"
```

```
sorm-report-connection-ori-aaa OID::= "12"
sorm-report-connection-ori-streams-content OID::= "13"
sorm-report-connection-ori-cert OID::= "14"
sorm-report-connection-ori-keys OID::= "15"

--- Абоненты
sorm-request-abonent-ori OID ::= "245"
sorm-report-abonent-ori OID ::= "70"

--- Справочники
sorm-request-dictionaries OID ::= "240"
sorm-report-dictionary-telcos OID::= "112"
sorm-report-dictionary-ori-services OID ::= "117"
sorm-report-dictionary-ori-events OID ::= "118"
sorm-report-dictionary-ori-resources OID ::= "119"
sorm-report-dictionary-ori-user-types OID ::= "130"
sorm-report-dictionary-ori-payment-services OID ::= "131"

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

sorm-report-presense-dictionaries OID::= "123"
sorm-report-presense-abonents-ori OID ::= "125"
sorm-report-presense-connections-ori OID ::= "126"

END
```

Dictionaries.asn

```
Dictionaries DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS DictionaryTask,
 DictionaryReport;

IMPORTS DateAndTime
 FROM Sorm
TAGGED,
 sorm-request-dictionaries,
 sorm-report-dictionary-telcos,
 sorm-report-dictionary-ori-services,
 sorm-report-dictionary-ori-events,
 sorm-report-dictionary-ori-resources,
 sorm-report-dictionary-ori-user-types,
 sorm-report-dictionary-ori-payment-services
FROM Classification;

--- 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
отчета)
}

--- DATA ObjectDescriptor принимает значение одно из:
--- sorm-report-dictionary-telcos
```

```
--- sorm-report-dictionary-ori-services
--- sorm-report-dictionary-ori-events
--- sorm-report-dictionary-ori-resources
--- sorm-report-dictionary-ori-user-types
--- sorm-report-dictionary-ori-payment-services

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

DictionaryRecordsVariants TAGGED ::= {
  telcosRecords --- коммуникационные интернет-сервисы ИС ОРИ
 | oriServices  ---  виды  информационных  сервисов,  предоставляемых  ОРИ  для
пользователей
 | oriEvents --- виды событий, регистрируемых ИС ОРИ при взаимодействии ИС ОРИ
с пользователем
 | oriResources --- типы информационных ресурсов, создаваемых пользователями в
ИС ОРИ
 | oriUserTypes --- типы пользователей, обслуживаемых ИС ОРИ
 |  oriPaymentServices  ---  виды  платежных  услуг  (платежных  сервисов),
используемых ОРИ
}

--- коммуникационные интернет-сервисы ИС ОРИ
telcosRecords TAGGED::= {
 OID {sorm-report-dictionary-telcos}
 DATA SEQUENCE OF TelcosRecord
}

TelcosRecord::= SEQUENCE {
 telco-id TelcoID, --- идентификатор ИС ОРИ
 begin-time DateAndTime, --- время начала действия
 end-time DateAndTime OPTIONAL, --- время конца действия
 description  UTF8String  (SIZE  (1  ..  256)),  ---  описание  (наименование)
коммуникационного интернет-сервиса ИС ОРИ
 mcc [0] NumericString (SIZE (3)) OPTIONAL, --- код страны
 mnc [1] NumericString (SIZE (3)) OPTIONAL - код ОРИ
}

--- виды информационных сервисов, предоставляемых ОРИ для пользователей
oriServices TAGGED ::= {
 OID {sorm-report-dictionary-ori-services}
 DATA SEQUENCE OF OriServiceRecord
}

OriServiceRecord ::= SEQUENCE {
 telco-id TelcoID, --- идентификатор ИС ОРИ
 service-id INTEGER (0 .. 65535), --- идентификатор сервиса
 description UTF8String (SIZE (1 .. 256)), --- описание
 begin-time DateAndTime, --- время начала действия
 end-time DateAndTime OPTIONAL --- время конца действия
}

--- виды событий, регистрируемых ИС ОРИ при взаимодействии ИС ОРИ с
пользователем
oriEvents TAGGED ::= {
 OID {sorm-report-dictionary-ori-events}
 DATA SEQUENCE OF OriEventRecord
}
```

```
OriEventRecord ::= SEQUENCE {
 telco-id TelcoID, --- идентификатор ИС ОРИ
 service-id INTEGER (0 .. 65535), --- идентификатор сервиса
 event-id INTEGER (0 .. 65535), --- идентификатор события
 description UTF8String (SIZE (1 .. 256)), --- описание
 begin-time DateAndTime, --- время начала действия
 end-time DateAndTime OPTIONAL --- время конца действия
}

--- типы информационных ресурсов, создаваемых пользователями в ИС ОРИ
oriResources TAGGED ::= {
 OID {sorm-report-dictionary-ori-resources}
 DATA SEQUENCE OF OriResourceRecord
}

OriResourceRecord ::= SEQUENCE {
 telco-id TelcoID, --- идентификатор ИС ОРИ
 service-id INTEGER (0 .. 65535), --- идентификатор сервиса
 resource-id INTEGER (0 .. 65535), --- идентификатор ресурса
 description UTF8String (SIZE (1 .. 256)), --- описание
 begin-time DateAndTime, --- время начала действия
 end-time DateAndTime OPTIONAL --- время конца действия
}

--- типы пользователей, обслуживаемых ИС ОРИ
oriUserTypes TAGGED ::= {
 OID {sorm-report-dictionary-ori-user-types}
 DATA SEQUENCE OF OriUserTypeRecord
}

OriUserTypeRecord ::= SEQUENCE {
 telco-id TelcoID, --- идентификатор ИС ОРИ
 service-id INTEGER (0 .. 65535), --- идентификатор сервиса
 user-type-id INTEGER (0 .. 65535), --- тип пользователя
 description UTF8String (SIZE (1 .. 256)), --- описание
 begin-time DateAndTime, --- время начала действия
 end-time DateAndTime OPTIONAL --- время конца действия)
}

--- виды платежных услуг (платежных сервисов), используемых ОРИ
oriPaymentServices TAGGED ::= {
 OID {sorm-report-dictionary-ori-payment-services}
 DATA SEQUENCE OF OriPaymentServiceRecord
}

OriPaymentServiceRecord ::= SEQUENCE {
 telco-id TelcoID, --- идентификатор ИС ОРИ
 service-id INTEGER (0 .. 65535), --- идентификатор сервиса
 payment-service-id INTEGER (0 .. 65535), --- идентификатор платежной услуги
 description UTF8String (SIZE (1 .. 256)), --- описание
 begin-time DateAndTime, --- время начала действия
 end-time DateAndTime OPTIONAL --- время конца действия
}

END
```

IdentifiersORI.asn

```
IdentifiersORI DEFINITIONS IMPLICIT TAGS::=
BEGIN

EXPORTS UserID,
  UserIdentifier,
```

```
  UserTechnicalIdentifier,
  ResourceIdentifier,
  ResourceName,
  ORIStreamID;
UserIdentifier::= SEQUENCE {
  user-id [0] UserID, --- идентификатор пользователя в системе (имя
пользователя)
  user-type [1] INTEGER (0.. 65535) --- тип пользователя (расшифровывается по
справочнику)
}

UserTechnicalIdentifier::= CHOICE {
 ip-address [0] UserIPAddress, --- ip-адрес и порт
 msisdn [1] UTF8String (SIZE (1.. 64)), --- номер телефона
 email [2] UTF8String (SIZE (1.. 256)), --- электронная почта
 program-name [3] UTF8String (SIZE (1.. 1024)), --- имя программы клиента
 other [4] UTFSString (SIZE (1.. 1024)) --- прочая техническая информация
}

UserIPAddress::= SEQUENCE {
 ip-address [0] UTF8String (SIZE (1.. 64)), --- ip-адрес
 ip-port [1] INTEGER (0.. 65535) OPTIONAL --- TCP/UDP порт
}

UserID::= UTF8String (SIZE (1.. 128))

ResourceIdentifier::= SEQUENCE {
 resource-name [0] ResourceName, --- наименование ресурса
 resource-type[1]INTEGER(0..65535)OPTIONAL --- тип ресурса (расшифровывается по
справочнику)
}

ResourceName::= UTF8String (SIZE (1.. 4096))

ORIStreamID::= UTF8String (SIZE (1 .. 1024))

END
```

Locations.asn

```
Locations DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS LocationInfo;

IMPORTS ProjectionType
  FROM NetworkIdentifiers;

MobileORILocation ::= SEQUENCE {
 mcc [0] UTF8String (SIZE (1 .. 4)) OPTIONAL, --- код страны оператора связи
пользователя ОРИ
 mnc [1] UTF8String (SIZE (1 .. 3)) OPTIONAL, --- код оператора связи
пользователя ОРИ
 lac [2] INTEGER (0 .. 65535) OPTIONAL, --- код LAC оператора связи пользователя
ОРИ
 cell [3] INTEGER (0 .. 4294967295) OPTIONAL, --- код БС оператора связи
пользователя ОРИ
 network-operator [4] UTF8String (SIZE (1 .. 64)) OPTIONAL --- наименование
оператора связи пользователя ОРИ
}

GeoLocation ::= SEQUENCE {
 latitude-grade REAL, --- широта
```

```
 longitude-grade REAL, --- долгота
 projection-type ProjectionType --- тип проекции координат
}

LocationInfo ::= SEQUENCE {
 mobile-ori-location [0] MobileORILocation OPTIONAL,
 geo-location [1] GeoLocation OPTIONAL,
 description [2] UTF8String (SIZE (1 .. 4096)) OPTIONAL
}

END
```

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 СОРМ"
ManagementRequest::= CHOICE {
 get-structure [0] GetStructureRequest, --- запрос на получение структуры ИС
СОРМ - КТС и модулей 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 ИС СОРМ - КТС и модулей СПО
GetStructureRequest::= NULL

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

RequestedHardwareModules::= SEQUENCE OF ModuleId

RequestedSoftwareModules::= SEQUENCE OF ModuleId

--- 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 модуля 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

--- тип сообщения "ответ на команду управления ИС СОРМ"
ManagementResponse ::= CHOICE {
 get-structure [0] GetStructureResponse, --- ответ на запрос получения
структуры ИС СОРМ - 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 ИС СОРМ - 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 ИС СОРМ
}

--- 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 ИС СОРМ
}

--- 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
```

```
}

SormHardwareModules ::= SEQUENCE OF SormHardwareModule

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
}

HwParameterGroups ::= SEQUENCE OF HwParameterGroup

HwParameterGroup ::= SEQUENCE {
 group-name UTF8String (SIZE (1 .. 512)), --- наименование группы параметров
для КТС
 module-parameters ModuleParameters --- перечень параметров для КТС
}

SormSoftwareModules ::= SEQUENCE OF SormSoftwareModule

SormSoftwareModule ::= SEQUENCE {
 module-id ModuleId, --- уникальный идентификатор данного модуля
 hardware-module-id ModuleId, --- идентификатор КТС, на котором работает 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 SormSoftwareModules OPTIONAL --- субмодули
}

ModuleParameters ::= SEQUENCE OF ModuleParameter

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

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

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

END
```

NetworkIdentifiers.asn

```
NetworkIdentifiers DEFINITIONS IMPLICIT TAGS ::=
BEGIN
EXPORTS NetworkPeerInfo,
  ProjectionType;

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

```
--- IP-адрес
IPAddress ::= CHOICE {
 ipv4 [0] IPV4Address, --- IPv4-адрес
 ipv6 [1] IPV6Address --- IPv6-адрес
}

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

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

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

--- тип проекции координат
ProjectionType ::= ENUMERATED {
 wgs84 (0),
 utm (1),
 sgs85 (2)
}

END
```

Reports.asn

```
Reports DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS reportMessage,
  Acknowledgement;
IMPORTS TAGGED,
  sorm-message-report
  FROM Classification

  MessageID
  FROM Sorm

  TaskID
  FROM Tasks

  DictionaryReport
  FROM Dictionaries
  PresenseReport
  FROM ReportsPresense
  AbonentsORIReport
  FROM ReportsAbonentsORI

  ConnectionsORIReport
  FROM ReportsConnectionsORI;

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 отчет
```

```
 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 ReportDataBiock --- блок данных отчета
}

ReportDataBlock ::= CHOICE {
 dictionary [0] DictionaryReport, --- отчеты задач пополнения справочников
(нормативно-справочная информация)
 presense [6] PresenseReport, --- отчеты задач по запросу наличия в ИС СОРМ
информации
 abonents-ori [8] AbonentsORIReport, --- отчеты задач по принадлежности
абонентов организаторов распространения
информации
 connections-ori [9] ConnectionsORIReport --- отчеты задач по соединениям
абонентов организаторов распространения
информации
}

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

END
```

ReportsAbonentsORI.asn

```
ReportsAbonentsORI DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS AbonentsORIReport;

IMPORTS TAGGED,
  sorm-report-abonent-ori
  FROM Classification

  DateAndTime
  FROM Sorm

  TelcoID
  FROM Tasks

  UserID,
  UserIdentifier
  FROM IdentifiersORI;

AbonentsORIReport ::= SEQUENCE {
id TAGGED.&id ({AbonentsORIReportVariants}),
data TAGGED.&Data ({AbonentsORIReportVariants}{@id})
}

AbonentsORIReportVariants TAGGED ::= {
 reportAbonentORI
}

reportAbonentORI TAGGED ::= {
```

```
 OID {sorm-report-abonent-ori}
 DATA SEQUENCE OF AbonentsORIRecord
}

AbonentsORIRecord ::= SEQUENCE {
 telco-id TelcoID, --- идентификатор ИС ОРИ,
 user-identifier UserIdentifier, --- идентификатор пользователя
 datetime-registered DateAndTime, --- дата и время регистрации
 abonent-info AbonentInfoORI, --- информация о пользователе
 abonent-contacts [0] AbonentContactsORI OPTIONAL, --- контактные данные
пользователя
 im-identifiers [1] SEQUENCE OF AbonentImIdentifierORI OPTIONAL, ---
идентификаторы пользователя в других
средствах электронного взаимодействия
 datetime-updated [2] DateAndTime OPTIONAL, --- дата и время обновления
информации
 datetime-unregistered [3] DateAndTime OPTIONAL, --- дата и время прекращения
регистрации
 contract-date [4] DateAndTime OPTIONAL, --- дата и время заключения договора
 contract [5] UTF8String (SIZE (1 .. 64)) OPTIONAL, --- номер договора
 additional [6] SEQUENCE OF AdditionalInfo OPTIONAL --- дополнительная
информация о пользователе
 service-id [7] INTEGER (0 .. 65535) OPTIONAL, --- идентификатор сервиса
(расшифровывается по справочнику)
}

--- информация о пользователе
AbonentInfoORI ::= SEQUENCE {
 nick-name [0] UTF8String (SIZE (1 .. 1024)) OPTIONAL, --- псевдоним
пользователя
 birth-date GeneralizedTime OPTIONAL, --- дата рождения
 address [1] ReportedAddressORI OPTIONAL, --- адресные данные
 name-info [2] ReportedNameInfoORI OPTIONAL, --- ФИО
 passport-info [3] ReportedPassportInfoORI OPTIONAL, --- паспортные данные
 langs [4] SEQUENCE OF UTF8String (SIZE (1 .. 64)) OPTIONAL, --- список языков,
которыми владеет пользователь
 relatives [5] SEQUENCE OF UserID OPTIONAL --- список родственников пользователя
}

--- адресные данные
ReportedAddressORI::= CHOICE {
 struct-info [0] ReportedStructAddressORI,
 unstruct-info [1] UTF8String (SIZE (1.. 1024))
}

ReportedStructAddressORI::= SEQUENCE {
 country [0] UTF8String (SIZE (1.. 128)) OPTIONAL, --- страна
 region (1) UTF8String (SIZE (1.. 128)) OPTIONAL, --- область
 city [2] UTF8String (SIZE (1.. 128)) OPTIONAL, --- город, поселок, деревня
 street [3] UTF8String (SIZE (1.. 128)) OPTIONAL, --- улица
 building [4] UTF8String (SIZE (1.. 128)) OPTIONAL, --- дом, строение
 apartment [5] UTF8String (SIZE (1.. 128)) OPTIONAL --- квартира, офис
}

--- ФИО
ReportedNameInfoORI::= CHOICE {
 struct-info [0] ReportedStructNameInfoORI,
 unstruct-info [1] UTF8String (SIZE (1.. 1024))
}

ReportedStructNameInfoORI::= SEQUENCE {
 given-name [0] UTF8String (SIZE (1.. 128)) OPTIONAL, --- имя
 initial [1] UTF8String (SIZE (1.. 128)) OPTIONAL, --- отчество (при наличии)
 family-name [2] UTF8String (SIZE (1.. 128)) OPTIONAL --- фамилия
```

```
}

--- паспортные данные
ReportedPassportInfoORI::= CHOICE {
 struct-info [0] ReportedStructPassroptInfoORI,
 unstruct-info [1] UTF8String (SIZE (1.. 1024))
}

ReportedStructPassroptInfoORI::= SEQUENCE {
passport-serial [0] UTF8String (SIZE (1..16)) OPTIONAL, --- серия паспорта
passport-number [1] UTF8String (SIZE (1..16)) OPTIONAL --- номер паспорта
}

--- контактные данные пользователя
AbonentContactsORI::= SEQUENCE {
 msisdn [0] SEQUENCE OF UTF8String (SIZE (1.. 64)) OPTIONAL, --- список номеров
телефона пользователя
 email [1] SEQUENCE OF UTF8String (SIZE (1.. 256)) OPTIONAL --- список адресов
электронной почты пользователя
}

--- идентификаторы пользователя в других средствах электронного взаимодействия
AbonentImIdentifierORI::= SEQUENCE {
 service-name UTF8String (SIZE (1.. 128)), --- наименование сервиса
 service-id UTF8String (SIZE (1.. 256)) --- идентификатор пользователя в сервисе
}

--- дополнительная информация о пользователе
AdditionalInfo::= SEQUENCE {
 title UTFSString (SIZE (1.. 1024)), --- наименование о дополнительных сведениях
 content UTF8String (SIZE (1.. 4096)) --- содержание дополнительных сведений
}

END
```

ReportsConnectionsORI.asn

```
ReportsConnectionsORI DEFINITIONS IMPLICIT TAGS::=
BEGIN

EXPORTS ConnectionsORIReport,
  reportConnectionsDataORI,
  reportConnectionsLinkORI,
  reportConnectionsAAAORI,
  reportConnectionsStreamsContentORI,
  reportConnectionsCertORI,
  reportConnectionsKeysORI;

IMPORTS TAGGED,
  sorm-report-connection-ori-data,
  sorm-report-connection-ori-link,
  sorm-report-connection-ori-aaa,
  sorm-report-connection-ori-streams-content,
  sorm-report-connection-ori-cert,
  sorm-report-connection-ori-keys
  FROM Classification

  DateAndTime
  FROM Sorm

  TelcoID
  FROM Tasks
```

```
  UserID,
  UserIdentifier,

  UserTechnicalIdentifier,
  ResourceIdentifier,
  ORIStreamID
  FROM IdentifiersORI

  NetworkPeerInfo
  FROM NetworkIdentifiers

  LocationInfo
  FROM Locations;
ConnectionsORIReport::= SEQUENCE {
 id TAGGED.&id ({ReportedConnectionsORIVariants}),
 data TAGGED.&Data ({ReportedConnectionsORIVariants}{@id})
}
ReportedConnectionsORIVariants TAGGED::= {
  reportConnectionsDataORI   -   события   информационного   взаимодействия
пользователей в ИС ОРИ
 |  reportConnectionsLinkORI  ---  события  добавления/исключения  связанных
пользователей
 | reportConnectionsAAAORI --- события регистрации, прекращения регистрации,
авторизации, выхода
из информационного сервиса
 |  reportConnectionsStreamsContentORI  -  содержимое  изображений,  звуков,
голосовой информации, видео- и иных
электронных сообщений
 | reportConnectionsCertORI
 | reportConnectionsKeysORI
}

--- события информационного взаимодействия пользователей в ИС ОРИ
reportConnectionsDataORI TAGGED::= {
 OID {sorm-report-connection-ori-data}
 DATA SEQUENCE OF ConnectionDataORIRecord
}

ConnectionDataORIRecord::= SEQUENCE {
 telco-id TelcoID, - идентификатор ИС ОРИ
 arrive-datetime DateAndTime, --- дата и время поступления информации
 datetime DateAndTime, --- дата и время
 service-id INTEGER (0.. 65535), --- идентификатор сервиса (расшифровывается по
справочнику)
 abonent-identifier UserIdentifier, --- идентификатор пользователя
 abonent-technical-identifier   SEQUENCE   OF   UserTechnicalIdentifier,   -
технические данные, идентифицирующие
пользователя
 event-id INTEGER (0.. 65535), - тип события (расшифровывается по справочнику)
 contacts-identifier SEQUENCE OF UserIdentifier, --- идентификаторы контактов
 abonent-location [0] Locationinfo OPTIONAL, --- местоположение пользователя
 resource-identifier [1] ResourceIdentifier OPTIONAL, --- ресурс
 resource-info [2] ResourceMetadataInfoORI OPTIONAL, --- техническая информация
о ресурсе
 owner-identifier  [3]  UserIdentifier  OPTIONAL,  ---  идентификатор  владельца
ресурса
 message-text [4] UTF8String OPTIONAL, --- текст сообщения без разметки и иной
служебной коммуникационной
информации
 payment-id [5] PaymentInfoORI OPTIONAL, --- техническая информация о платеже
 stream-id [6] ORIStreamID OPTIONAL --- идентификатор содержимого
}

ResourceMetadataInfoORI::= CHOICE {
```

```
 files-metadata [0] SEQUENCE OF FileMetadata,
 stream-metadata [1] StreamMetadata
}

--- техническая информация о файловых данных
FileMetadata::= SEQUENCE {
 filename UTF8String (SIZE (1.. 256)),
 filesize INTEGER,
 file-type UTF8String (SIZE (1.. 256)) OPTIONAL --- MIME-тип данных для файла
}

 --- техническая информация о потоковых данных
StreamMetadata::= SEQUENCE {
 timestamp [0] GeneralizedTime OPTIONAL,
 duration [1] INTEGER OPTIONAL,
 start-offset [2] INTEGER OPTIONAL,
 stop-offset [3] INTEGER OPTIONAL
}

PaymentInfoORI::= SEQUENCE {
 payment-identifier UTF8String (SIZE (1.. 128)), --- идентификатор транзакции,
счета, кошелька
 payment-service-id INTEGER (0.. 65535), --- идентификатор платежного сервиса
(расшифровывается по справочнику)
 payment-info UTF8String (SIZE (1.. 4096)) OPTIONAL --- дополнительная
информация
}

--- события добавления/исключения связанных пользователей
reportConnectionsLinkORI TAGGED::= {
 OID {sorm-report-connection-ori-link}
 DATA SEQUENCE OF ConnectionLinkORIRecord
}

ConnectionLinkORIRecord::= SEQUENCE {
telco-id TelcoID, - идентификатор ИС ОРИ,
 arrive-datetime DateAndTime, --- дата и время поступления информации
 datetime DateAndTime, - дата и время
 service-id INTEGER (0.. 65535), --- идентификатор сервиса (расшифровывается по
справочнику)
 abonent-id UserIdentifier, --- идентификатор пользователя, устанавливающего
связь
 event-id INTEGER (0.. 65535), --- тип события (расшифровывается по справочнику)
 linked-identifier LinkedIdentifier - идентификатор объекта, с которым
установлена связь
}

LinkedIdentifier::= CHOICE {
 connected-user-identifier [0] UserIdentifier, --- идентификатор пользователя
 connected-resource-identifier [1] ResourceIdentifier --- ресурс
}

--- события регистрации, прекращения регистрации, авторизации, выхода из
информационного сервиса
reportConnectionsAAAORI TAGGED::= {
 OID {sorm-report-connection-ori-aaa}
 DATA SEQUENCE OF ConnectionAAAORIRecord
}

ConnectionAAAORIRecord::= SEQUENCE {
 telco-id TelcoID, - идентификатор ИС ОРИ,
 arrive-datetime DateAndTime, --- дата и время поступления информации
 datetime DateAndTime, --- дата и время
```

```
 service-id INTEGER (0.. 65535), --- идентификатор сервиса (расшифровывается по
справочнику)
 abonent-identifier UserIdentifier, --- идентификатор пользователя
 abonent-technical-identifier    SEQUENCE    OF    UserTechnicalIdentifier,    -
технические данные, идентифицирующие пользователя
 event-id INTEGER (0.. 65535), --- тип события (расшифровывается по справочнику)
 abonent-location [0] LocationInfo OPTIONAL --- местоположение пользователя
}
--- содержимое изображений, звуков, голосовой информации, видео- и иных
электронных сообщений
reportConnectionsStreamsContentORI TAGGED::= {
 OID {sorm-report-connection-ori-streams-content}
 DATA ConnectionsStreamsContentORIRecords
}

ConnectionsStreamsContentORIRecords::= CHOICE {
 flow-data [0] SEQUENCE OF ConnectionsStreamsContentORIRecordFlow,
 file-data [1] SEQUENCE OF ConnectionsStreamsContentORIRecordFile
}

ConnectionsStreamsContentORIRecordFlow::= SEQUENCE {
 data OCTET STRING (SIZE (1..1048576)), --- содержимое блока
 codec UTF8String (SIZE (1.. 4096)) OPTIONAL, --- описание способа кодирования
в формате SDP в соответствии
с форматом RFC 2327;
 direction StreamsContentORIDirection OPTIONAL --- направление передачи
}

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

ConnectionsStreamsContentORIRecordFile::= OCTET STRING (SIZE (1.. 1048576))

reportConnectionsCertORI TAGGED::= {
 OID {sorm-report-connection-ori-cert}
 DATA SEQUENCE OF ConnectionCertORIRecord
}

ConnectionCertORIRecord::= SEQUENCE {
 telco-id TelcoID,
 arrive-datetime DateAndTime,
 datetime-from DateAndTime,
 datetime-to DateAndTime OPTIONAL,
 client NetworkPeerInfo,
 server NetworkPeerInfo,
 client-random OCTET STRING (SIZE (1.. 512)),
 pre-master-key OCTET STRING (SIZE (1.. 512)),
 server-random [0] OCTET STRING (SIZE (1.. 512)) OPTIONAL
}

reportConnectionsKeysORI TAGGED::= {
 OID {sorm-report-connection-ori-keys}
 DATA SEQUENCE OF ConnectionKeysORIRecord
}

ConnectionKeysORIRecord::= SEQUENCE {
 telco-id TelcoID,
 arrive-datetime DateAndTime,
 datetime-from DateAndTime,
 datetime-to [0] DateAndTime OPTIONAL,
 as-server [1] INTEGER OPTIONAL,
 as-client [2] INTEGER OPTIONAL,
```

```
 client NetworkPeerInfo,
 server NetworkPeerInfo,
 client-random OCTET STRING (SIZE (1.. 512)),
 pre-master-key OCTET STRING (SIZE (1.. 512)),
 server-random [0] OCTET STRING (SIZE (1.. 512)) OPTIONAL,
 sni [1] UTF8String (SIZE (1.. 128)) OPTIONAL,
 user-id [2] UTF8String (SIZE (1.. 64)) OPTIONAL,
 user-login [3] UTF8String (SIZE (1.. 64)) OPTIONAL,
 key-exchange [4] UTF8String (SIZE (1.. 16)) OPTIONAL,
 chipher-suite [5] INTEGER OPTIONAL,
 client-region [6] UTF8String (SIZE (1.. 10)) OPTIONAL,
 server-region [7] UTF8String (SIZE (1.. 10)) OPTIONAL,
 service-name [8] UTF8String (SIZE (1.. 16)) OPTIONAL
}

END
```

ReportsPresense.asn

```
ReportsPresense DEFINITIONS IMPLICIT TAGS::=
BEGIN

EXPORTS PresenseReport;

IMPORTS TAGGED,

  sorm-report-presense-dictionaries,
  sorm-report-presense-abonents-ori,
  sorm-report-presense-connections-ori
  FROM Classification
  FindRange
  FROM Sorm

  TelcoID
  FROM Tasks;

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

ReportedPresensesVariants TAGGED::= {
 dictionariesPresence --- отчет по наличию информации по справочникам
 | abonentsORIPresence --- отчет по наличию информации по данным о пользователях
ОРИ и их идентификаторам
 | connectionsORIPresence --- отчет по наличию информации по соединениям
пользователя ОРИ
}
--- отчет по наличию информации по справочникам
--- если какой-либо из справочников не публикуется ИС СОРМ, запись о нем
отсутствует
dictionariesPresence TAGGED::= {
 OID {sorm-report-presense-dictionaries}
 DATA SEQUENCE OF DictionaryInfo
}

--- запись отчета о наличии справочной информации
DictionaryInfo::= SEQUENCE {
 telco-id TelcoID, --- идентификатор ИС ОРИ
 dict ObjectDescriptor, - тип справочника, по которому есть информация
 count INTEGER (1.. 4294967295), --- количество записей в справочнике
```

```
 change-dates FindRange --- минимальная и максимальная дата, минимальное и
максимальное время изменения
записей в справочнике
}

--- отчет по наличию информации по данным о пользователях ОРИ и их
идентификаторам
abonentsORIPresence TAGGED::= {
 OID {sorm-report-presense-abonents-ori}
 DATA SEQUENCE OF AbonentsORIPresenseRecord
}

AbonentsORIPresenseRecord::= SEQUENCE {
 telco-id TelcoID, --- идентификатор ИС ОРИ
 range FindRange, --- интервал времени, на который имеются данные в ИС ОРИ
 count INTEGER, --- количество записей
 service-id INTEGER (0.. 65535) OPTIONAL --- идентификатор сервиса
}

--- отчет по наличию информации по соединениям пользователя ОРИ
connectionsORIPresence TAGGED::= {
 OID {sorm-report-presense-connections-ori}
 DATA SEQUENCE OF ConnectionsORIPresenseRecord
}

ConnectionsORIPresenseRecord::= SEQUENCE {
 telco-id TelcoID, --- идентификатор ИС ОРИ
 range FindRange, --- интервал времени, на который имеются данные в ИС ОРИ
 count INTEGER, --- количество записей
 data-type ConnectionsORIPresenseType, --- вид событий в ИС ОРИ, информация по
которым есть в ИС СОРМ
 service-id INTEGER (0.. 65535) OPTIONAL --- идентификатор сервиса
}

ConnectionsORIPresenseType::= ENUMERATED {
 data (0), --- записи об информационном взаимодействии пользователей в ИС ОРИ
 link (1), --- записи о добавлении/исключении связанных пользователей
 aaa (2), - записи о регистрации, прекращения регистрации, авторизации, выхода
из информационного сервиса
 certs (3),
 keys (4)
}

END
```

ReportsUNI.asn

```
ReportsUNI DEFINITIONS IMPLICIT TAGS::=
BEGIN

EXPORTS reportMessageUNI;

IMPORTS TAGGED,
  sorm-message-report-uni
  FROM Classification

  ObjectUNI
  FROM Tasks

  Acknowledgement
  FROM Reports

   reportConnectionsDataORI,
```

```
   reportConnectionsLinkORI,
   reportConnectionsAAAORI,
   reportConnectionsStreamsContentORI
   FROM ReportsConnectionsORI;
 reportMessageUNI TAGGED::= {
  OID {sorm-message-report-uni}
  DATA CHOICE {
   report [0] ReportUNI, - тип сообщения "отчет"
   ack [1] Acknowledgement - тип сообщения "подтверждение"
 }
}

--- Блок данных сообщения типа "отчет"
ReportUNI::= SEQUENCE {
 object-unis SEQUENCE OF ObjectUNI, --- идентификаторы объектов наблюдения, по
которым отобраны данные
 report-block ReportBlockUNI - блок данных отчета
}

ReportBlockUNI::= CHOICE {
 statistics [0] StatisticsReportedConnections,
 content [1] ContentReportBlockUNI
}

StatisticsReportedConnections::= SEQUENCE {
 id TAGGED.&id ({StatisticsReportedConnectionsVariants}),
 data TAGGED.&Data ({StatisticsReportedConnectionsVariants}{@id})
}

StatisticsReportedConnectionsVariants TAGGED::= {
  reportConnectionsDataORI   ---   события   информационного   взаимодействия
пользователей в ИС ОРИ
 |  reportConnectionsLinkORI  ---  события  добавления/исключения  связанных
пользователей
 | reportConnectionsAAAORI --- события регистрации, прекращения регистрации,
авторизации, выхода
из информационного сервиса
}

ContentReportBlockUNI ::= SEQUENCE {
 report-block ContentReportedConnections, --- блок данных отчета
 stream-id ORIStreamID, --- идентификатор содержимого
 last-block BOOLEAN --- признак последнего блока
}

ContentReportedConnections ::= SEQUENCE {
 id TAGGED.&id ({ContentReportedConnectionsVariants}),
 data TAGGED.&Data ({ContentReportedConnectionsVariants}{@id})
}

ContentReportedConnectionsVariants TAGGED ::= {
 reportConnectionsStreamsContentORI   ---   содержимое   изображений,   звуков,
голосовой  информации,
видео- и иных электронных сообщений
}

END
```

RequestedAbonentsORI.asn

```
RequestedAbonentsORI DEFINITIONS IMPLICIT TAGS ::=
BEGIN
```

```
EXPORTS RequestedAbonentsORI;

IMPORTS TAGGED,
  sorm-request-abonent-ori
  FROM Classification
  UserID
  FROM IdentifiersORI;

RequestedAbonentORI := SEQUENCE {
 id TAGGED.&id ({RequestedAbonentORIVariants}),
 data TAGGED.&Data ({RequestedAbonentORIVariants}(@id})
}

RequestedAbonentORIVariants TAGGED ::= {
 requestedORIabonent
}

requestedORIabonent TAGGED ::= {
 OID {sorm-request-abonent-ori}
 DATA CHOICE {
  user-id [0] UserID, --- идентификатор пользователя
  nick-name [1] UTF8String (SIZE (1 .. 1024)), --- псевдоним пользователя
  given-name [2] UTF8String (SIZE (1 .. 128)), --- имя
  initial [3] UTF8String (SIZE (1 .. 128)), --- отчество (при наличии)
  family-name [4] UTF8String (SIZE (1 .. 128)), --- фамилия
  address [5] RequestedAddressORI, --- адресные данные
  passport [6] RequestedPassportORI, --- паспортные данные
  relatives [7] UserID, --- идентификатор родственника
  msisdn [8] UTF8String (SIZE (1 .. 64)), --- номер телефона пользователя
  email [9] UTF8String (SIZE (1 .. 256)), --- электронная почта пользователя
  im-id [10] UTF8String (SIZE (1 .. 256)), --- идентификатор в сетях мгновенного
обмена сообщениями
  contract [11] UTF8String (SIZE (1 .. 64)) --- номер договора
 }
}

RequestedAddressORI ::= SEQUENCE {
 country [0] UTF8String (SIZE (1 .. 128)) OPTIONAL, --- страна
 region [1] UTF8String (SIZE (1 .. 128)) OPTIONAL, --- область
 city [2] UTF8String (SIZE (1 .. 128)) OPTIONAL, --- город, поселок, деревня,
населенный пункт
 street [3] UTF8String (SIZE (1 .. 128)) OPTIONAL, --- улица
 building [4] UTF8String (SIZE (1 .. 128)) OPTIONAL, --- дом, строение
 apartment [5] UTF8String (SIZE (1 .. 128)) OPTIONAL --- квартира, офис
}

RequestedPassportORI ::= SEQUENCE {
 passport-serial [0] UTF8String (SIZE (1 .. 16)) OPTIONAL, --- серия паспорта
 passport-number [1] UTF8String (SIZE (1 .. 16)) OPTIONAL --- номер паспорта
}

END
```

RequestedConnectionsORI.asn

```
RequestedConnectionsORI DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS RequestedConnectionORI;

IMPORTS TAGGED,
  sorm-request-connection-ori-data-standart,
  sorm-request-connection-ori-data-extended,
```

```
  sorm-request-connection-ori-link,
  sorm-request-connection-ori-aaa,
  sorm-request-connection-ori-streams-content,
  sorm-request-connection-ori-cert,
  sorm-request-connection-ori-keys
  FROM Classification

  UserID,
  UserTechnicalIdentifier,
  ResourceName,
  ORIStreamID
  FROM IdentifiersORI

  NetworkPeerInfo
  FROM NetworkIdentifiers

  LocationInfo
  FROM Locations;

RequestedConnectionORI ::= SEQUENCE {
 id TAGGED.&id ({RequestedConnectionORIVariants}),
 data TAGGED.&Data ({RequestedConnectionORIVariants}{@id})
}

RequestedConnectionORIVariants TAGGED ::= {
  requestedORIDataStandart   ---   события   информационного   взаимодействия
пользователей в ИС ОРИ
(базовый  набор критериев)
 |   requestedORIDataExtended   ---   события   информационного   взаимодействия
пользователей в ИС ОРИ
(расширенный  набор критериев)
 | requestedORILink --- события добавления/исключения связанных пользователей
 |   requestedORIAAA   ---   события   регистрации,   прекращения   регистрации,
авторизации, выхода из информационного
сервиса
 | requestedORIStreamsContent --- содержимое изображений, звуков, голосовой
информации, видео- и иных
электронных сообщений
 | requestedORICert
 | requestedORIKeys
}

--- события информационного взаимодействия пользователей в ИС ОРИ (базовый набор
критериев)
requestedORIDataStandart TAGGED ::= {
 OID {sorm-request-connection-ori-data-standart}
 DATA CHOICE {
  abonent-id [0] UserID, --- идентификатор пользователя
  abonent-technical-identifier  [1]  UserTechnicalIdentifier  ---  технические
данные, идентифицирующие
пользователя
 }
}

--- события информационного взаимодействия пользователей в ИС ОРИ (расширенный
набор критериев)
requestedORIDataExtended TAGGED ::= {
 OID {sorm-request-connection-ori-data-extended}
 DATA CHOICE {
  service-id   [0]   INTEGER   (0   ..   65535),   ---   идентификатор   сервиса
(расшифровывается по справочнику)
  abonent-id [1] UserID, --- идентификатор абонента
  abonent-technical-identifier  [2]  UserTechnicalIdentifier,  ---  технические
данные, идентифицирующие абонента
```

```
  abonent-location [3] LocationInfo, --- местоположение абонента
  contact-id [4] UserID, --- идентификатор контакта
  resource-name [5] ResourceName, --- наименование ресурса
  event-id [6] INTEGER (0 .. 65535), --- тип события (расшифровывается по
справочнику)
  owner-id [7] UserID, --- идентификатор владельца ресурса
  message-text [8] UTF8String, --- текст сообщения без разметки и иной служебной
коммуникационной информации
  payment-identifier [9] UTF8String(SIZE (1 .. 128)), --- идентификатор платежа
  payment-info [10] UTF8String (SIZE (1 .. 4096)) --- платежная информация
 }
}

--- события добавления/исключения связанных пользователей
requestedORILink TAGGED ::= {
 OID {sorm-request-connection-ori-link}
 DATA CHOICE {
  user-id [0] UserID, --- идентификатор пользователя, установившего связь
  connected-user-id [1] UserID, --- идентификатор пользователя, с которым
установили связь
  connected-resource-name [2] ResourceName --- наименование ресурса, с которым
установили связь
 }
}

--- события регистрации, прекращения регистрации, авторизации, выхода из
информационного сервиса
requestedORIAAA TAGGED ::= {
 OID {sorm-request-connection-ori-aaa}
 DATA CHOICE {
  service-id [0] INTEGER (0 .. 65535), --- идентификатор сервиса
(расшифровывается по справочнику)
  user-id [1] UserID, --- идентификатор пользователя
  user-technical-identifier [2] UserTechnicalIdentifier, --- технические
данные, идентифицирующие пользователя
  user-location [3] LocationInfo, --- местоположение пользователя
  event-id [4] INTEGER (0 .. 65535) --- тип события (расшифровывается по
справочнику)
 }
}

--- содержимое изображений, звуков, голосовой информации, видео- и иных
электронных сообщений
requestedORIStreamsContent TAGGED ::= {
 OID {sorm-request-connection-ori-streams-content}
 DATA ORIStreamsContentRequest
}

ORIStreamsContentRequest ::= SEQUENCE {
 stream-id ORIStreamID, --- идентификатор запрашиваемых данных
 duration [1] INTEGER OPTIONAL, --- длительность потоковых данных
 start-offset [2] INTEGER OPTIONAL, --- начальное  смещение потоковых данных
 stop-offset [3] INTEGER OPTIONAL --- конечное смещение потоковых данных
}

requestedORICert TAGGED ::= {
 OID {sorm-request-connection-ori-cert}
 DATA CHOICE {
  client [0] NetworkPeerInfo,
  server [1] NetworkPeerInfo
 }
}

requestedORIKeys TAGGED ::= {
```

```
 OID (sorm-request-connection-ori-keys)
 DATA CHOICE {
  as-server [0] INTEGER,
  as-client [1] INTEGER,
  client [2] NetworkPeerInfo,
  server [3] NetworkPeerInfo,
  sni [4] UTF8String (SIZE (1 .. 128)),
  user-id [5] UTF8String (SIZE (1 .. 64)),
  user-login [6] UTF8String (SIZE (1 .. 64)),
  client-region [7] UTF8String (SIZE (1 .. 10)),
  server-region [8] UTF8String (SIZE (1 .. 10)),
  service-name [9] UTF8String (SIZE (1 .. 16))
 }
}

END
```

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 на открытие 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, которое
может обеспечить ИС СОРМ); должно быть меньше или равно окну, переданному в
сообщении ConnectRequest
```

```
 confirmed-session-timeout INTEGER (60 .. 2592000), --- подтвержденное
максимальное время неактивности; должно
быть больше или равно значению времени, переданному в сообщении ConnectRequest
 confirmed-data-load-timeout INTEGER (1 .. 60), --- подтвержденный таймаут
начала передачи блоков отчетов; должен
быть больше или равен значению таймаута, переданному в сообщении ConnectRequest
 confirmed-request-response-timeout INTEGER (1 .. 60), --- подтвержденный
таймаут ответа на запрос; должен быть
больше или равен значению таймаута, переданному в сообщении ConnectRequest
 supports SEQUENCE OF ObjectDescriptor --- весь список поддерживаемых СОРМ типов
запросов, типов отчетов
}

--- согласование поддерживаемых типов со стороны ПУ
AdjustmentRequest ::= SEQUENCE {
 supports SEQUENCE OF ObjectDescriptor --- список поддерживаемых ПУ типов
запросов, типов отчетов; данный список
должен быть меньшим либо равным списку в сообщении ConnectRequest
}

--- ответ на запрос согласования данных
AdjustmentResponse ::= NULL

--- запрос на закрытие сессии
DisconnectRequest ::= NULL

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

END
```

Sorm.asn

```
Sorm DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS DateAndTime,
  FindRange,
  MessageID;

IMPORTS TAGGED
  FROM Classification

  sessionMessage
  FROM Sessions

  trapMessage
  FROM Traps

  taskMessage
  FROM Tasks

  reportMessage
  FROM Reports

  managementMessage
  FROM Management

  unformatted Message
  FROM Unformatted

  reportMessageUNI
  FROM ReportsUNI
```

```
Version ::= PrintableString
vers Version ::= "3.0.0" --- текущая версия протокола

--- Оболочка сообщения СОРМ
Message ::= SEQUENCE {
 version Version DEFAULT vers, --- 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}), --- идентификатор блока данных
 data TAGGED.&Data ({SormPDUs}{@id}) --- данные блока данных
}

--- Блок данных сообщения
SormPDUs TAGGED ::= {
  sessionMessage --- сообщения организации сессии
 | trapMessage --- сообщения сигналов
 | taskMessage --- сообщения работы с задачами
 | reportMessage --- сообщения работы с отчетами
 | managementMessage --- сообщения канала передачи мониторинга (КПМ)
 | unformattedMessage --- сообщения канала передачи неформатированных данных
(КПНФ)
 | reportMessageUNI --- сообщения с отобранными по объектам контроля данными
}

--- 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] DateAndTime OPTIONAL --- время и дата окончания поиска информации
}

END
```

Sorm743.set.asn

```
Classification.asn
Dictionaries.asn
IdentifiersORI.asn
Locations.asn
Management.asn
NetworkIdentifiers.asn
Reports.asn
ReportsAbonentsORI.asn
ReportsConnectionsORI.asn
ReportsPresense.asn
ReportsUNI.asn
RequestedAbonentsORI.asn
RequestedConnectionsORI.asn
Sessions.asn
Sorm.asn
Tasks.asn
TasksAbonentsORI.asn
TasksConnectionsORI.asn
TasksPresense.asn
TasksUNI.asn
```

Traps.asn
Unformatted.asn

Tasks.asn

```
Tasks DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS taskMessage,
  TaskID,
  TelcoID,
  TelcoList,
  LogicalOperation,
  CreateTaskResponse;

IMPORTS TAGGED,
  sorm-message-task
  FROM Classification

  FindRange,
  MessageID
  FROM Sorm

  DictionaryTask
  FROM Dictionaries

  PresenseTask
  FROM TasksPresense

  AbonentsORITask
  FROM TasksAbonentsORI

  ConnectionsORITask
  FROM TasksConnectionsORI;
  UNIControlTaskRequest,
  UNIControlTaskResponse
  FROM TasksUNI;

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
  uni-task-request [12] UNIControlTaskRequest, --- запрос на постановку/снятие
объекта наблюдения на контроль
  uni-task-response  [13]  UNIControlTaskResponse  ---  ответ  на  запрос
постановки/снятия объекта наблюдения
с контроля
 }
}
```

```
--- запрос готовности данных
DataReadyRequest ::= NULL

--- ответ на запрос готовности данных
DataReadyResponse ::= SEQUENCE OF DataReadyTaskRecord

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

TaskResult ::= SEQUENCE {
 result TaskStatus,
 report-records-number [0] INTEGER (0 .. 999999999999) OPTIONAL, --- для
выполненной задачи количество записей
в отчете
 report-limit-exeeded [1] BOOLEAN OPTIONAL, --- количество записей превысило
лимит, заданный при создании задачи
 error-description [2] UTF8String (SIZE (1 .. 256)) OPTIONAL --- краткое
описание произошедшей ошибки, если
обнаружена
}

TaskStatus ::= ENUMERATED {
 data-not-ready (0), --- данные не готовы, задача еще выполняется
 data-ready (1), --- данные есть, задача выполнена
 data-not-found (2), --- данных нет, задача выполнена
 error (3) --- в процессе выполнения задачи произошла ошибка
}

--- запрос загрузки данных
DataLoadRequest ::= TaskID
--- ответ на запрос загрузки данных
DataLoadResponse ::= SEQUENCE {
 task-id TaskID, --- идентификатор задачи, сгенерировавшей данный отчет
 data-exists BOOLEAN, ---- признак существования результатов исполнения
задачи (есть данные или нет)
 data-blocks-number INTEGER (0.. 999999999999) OPTIONAL, --- количество блоков
в отчете
 error-description UTF8String (SIZE (1 .. 256)) OPTIONAL --- краткое описание
ошибки, если обнаружена
}

--- запрос удаления данных
DataDropRequest ::= TaskID

--- ответ на запрос удаления данных
DataDropResponse ::= SEQUENCE {
 task-id TaskID, --- идентификатор задачи, данные которой будут удалены
 successful BOOLEAN, --- признак успешного выполнения запроса
 error-description UTF8String (SIZE (1 .. 256)) OPTIONAL --- краткое описание
ошибки, если обнаружена
}

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

--- ответ на запрос прерывания загрузки данных
DataInterruptResponse ::= SEQUENCE {
 request-id MessageID, --- идентификатор прерванного запроса загрузки данных
 successful BOOLEAN, --- признак успешного выполнения запроса
 data-blocks-available INTEGER (0 .. 999999999999) OPTIONAL, --- количество
оставшихся непереданными блоков
```

```
 error-description UTF8String (SIZE (1 .. 256)) OPTIONAL --- краткое описание
ошибки, если обнаружена
}

--- запрос на создание задачи по обработке информации
CreateTaskRequest ::= SEQUENCE {
 telcos  [0]  TelcoList  OPTIONAL, --- список ИС ОРИ (других обслуживаемых ОРИ)
 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)
 presense [6] PresenseTask, --- задачи предоставления сведений о наличии данных
 abonents-ori  [7]  AbonentsORITask,  ---  задачи  поисков  по  принадлежности
абонентов организаторов распространения
информации
 connections-ori  [8]  ConnectionsORITask  ---  задачи  поисков  по  соединениям
абонентов организаторов распространения
информации
},

 find-by-arrive-time BOOLEAN OPTIONAL --- режим поиска информации (true - по
дате и времени поступления;
false - по дате и времени события; по умолчанию - false)
}

--- ответ на запрос создания задачи
CreateTaskResponse ::= SEQUENCE {
 task-id TaskID OPTIONAL, --- идентификатор задачи
 successful BOOLEAN, --- признак успешного выполнения запроса
 error-description UTF8String (SIZE  (1 .. 256)) OPTIONAL --- краткое описание
ошибки, если обнаружена
}

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

--- идентификатор ОРИ или коммуникационного интернет-сервиса ИС ОРИ
TelcoID ::= INTEGER (0 .. 65535)

--- список ИС ОРИ (других обслуживаемых ОРИ)
TelcoList ::= SEQUENCE OF TelcoID

--- идентификатор объекта наблюдения
ObjectUNI ::= INTEGER (0 .. 4294967295)

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
```

TasksAbonentsORI.asn

```
TasksAbonentsORI DEFINITIONS IMPLICIT TAGS ::=
BEGIN
```

```
EXPORTS AbonentsORITask;

IMPORTS LogicalOperation
  FROM Tasks

  RequestedAbonentORI
  FROM RequestedAbonentsORI;

--- Заданный при поиске временной интервал применяется к дате и времени регистрации (datetime-registered)
AbonentsORITask ::= CHOICE {
 validate-abonents [0] ValidateAbonentsORITask
}

ValidateAbonentsORITask ::= RequestedAbonentORIIdentifiers
RequestedAbonentORIIdentifiers ::= SEQUENCE OF RequestedAbonentORIIdentifier

RequestedAbonentORIIdentifier ::= CHOICE {
 separator [0] LogicalOperation,
 find-mask [1] RequestedAbonentORI
}

END
```

TasksConnectionsORI.asn

```
TasksConnectionsORI DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS ConnectionsORITask;

IMPORTS LogicalOperation
  FROM Tasks

  RequestedConnectionORI
  FROM RequestedConnectionsORI;

ConnectionsORITask ::= CHOICE {
 validate-standart [0] ValidateStandartTask, --- используется для запросов стандартного класса сложности
(sorm-request- connection-ori-data-standart; sorm-request-connection-ori-link; sorm-request-connection-ori-aaa)
 validate-extended [1] ValidateExtendedTask, --- используется для расширенных запросов
(sorm-request-connection-ori- data-extended)
 validate-streams-content [2] ValidateStreamsContentTask, --- используется для получения содержимого изображений,
звуков, голосовой информации, видео- и иных электронных сообщений (sorm-request-connection-ori-streams-content)
 valide-keys [3] ValidateTaskF1
}

ValidateStandartTask ::= RequestedConnectionIdentifiersORI
ValidateExtendedTask ::= RequestedConnectionIdentifiersORI
ValidateStreamsContentTask ::= RequestedConnectionORI
ValidateTaskF1 ::= RequestedConnectionIdentifiersORI

RequestedConnectionIdentifiersORI ::= SEQUENCE OF
RequestedConnectionParameterORI

RequestedConnectionParameterORI ::= CHOICE {
 separator [0] LogicalOperation,
```

```
 find-mask [1] RequestedConnectionORI
}

END
```

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 RequestPresenseData
}

RequestPresenseData ::= ENUMERATED {
 dictionaries (3), --- запрос наличия справочников
 abonents-ori (5), --- запрос наличия информации по данным о пользователях ОРИ
и их идентификаторам
 connections-ori (6) --- запрос наличия информации по соединениям пользователя
ОРИ
}

END
```

TasksUNI.asn

```
TasksUNI DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS UNIControlTaskRequest,
    UNIControlTaskResponse;

IMPORTS TAGGED
  FROM Classification

  UserID,
  UserTechnicalIdentifier
  FROM IdentifiersORI

  TelcoList,
  ObjectUNI
  FROM Tasks;

UNIControlTaskRequest ::= CHOICE {
 create-uni [0] CreateUNIRequest, --- запрос на создание объекта наблюдения и
постановки его на контроль
 drop-uni [1] DropUNIRequest --- запрос на снятие объекта наблюдения с контроля
и удаление объекта наблюдения
```

```
 get-uni [2] GetUNIRequest --- запрос на получение списка объектов наблюдения
установленных на контроль
}

UNIControlTaskResponse ::= CHOICE {
 create-uni [0] CreateUNIResponse, --- ответ на запрос создания объекта
наблюдения и постановки его на контроль
 drop-uni [1] DropUNIResponse --- ответ на запрос снятия объекта наблюдения с
контроля и удаление объекта
наблюдения
 get-uni [2] GetUNIResponse --- ответ на запрос на получение списка объектов
наблюдения установленных на контроль
}

CreateUNIRequest ::= SEQUENCE {
 uni-id ObjectUNI, --- идентификатор объекта наблюдения, переданный ПУ
 uni-criteria UNIParameterORI, --- критерии отбора для объекта наблюдения
 content-load BOOLEAN, --- режим выгрузки содержимого (true - статистика и
содержимое; false - только статистика)
 telcos TelcoList OPTIONAL --- список ИС ОРИ (других обслуживаемых ОРИ)
}

UNIParameterORI ::= CHOICE {
 user-id [0] UserID, --- идентификатор пользователя в системе (имя пользователя)
 user-technical-identifier [1] UserTechnicalIdentifier, --- технические данные,
идентифицирующие пользователя
 resource-name [2] UTF8String (SIZE (1 .. 4096)) --- наименование ресурса
}

CreateUNIResponse ::= SEQUENCE {
 uni-successful BOOLEAN, --- признак успешной постановки объекта наблюдения на
контроль
 uni-error-description UTF8String (SIZE (1 .. 256)) OPTIONAL --- краткое
описание ошибки, если обнаружена
}

DropUNIRequest ::= ObjectUNI --- идентификатор объекта наблюдения для снятия с
контроля

DropUNIResponse::= SEQUENCE {
 uni-successful BOOLEAN, - признак успешного снятия объекта наблюдения с
контроля
 uni-error-description   UTF8String (SIZE (1.. 256)) OPTIONAL --- краткое
описание ошибки, если обнаружена
}

GetUNIRequest ::= NULL
GetUNIResponse ::= SEQUENCE OF UNIResponse
UNIResponse ::= SEQUENCE {
 uni-id ObjectUNI, --- идентификатор объекта наблюдения, переданный ПУ
 uni-criteria UNIParameterORI, --- критерии отбора для объекта наблюдения
 content-load BOOLEAN, --- режим выгрузки содержимого (true - статистика и
содержимое; false - только статистика)
 telcos TelcoList OPTIONAL - список филиалов ОРИ (других обслуживаемых ОРИ)
}

END
```

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    MessageID    OPTIONAL --- номер сообщения, к которому
относится
данный сигнал
}

TrapType ::= ENUMERATED {
 heartbeat (0), --- тестовый пакет
 restart-software (1), --- перезапуск ПО
 unauthorized-access (2), --- попытка несанкционированного доступа
 critical-error   (3), --- критическая ошибка ПО, потеря данных, дальнейшая
работа невозможна
 major-error  (4), --- серьезная ошибка ПО, потеря данных, но дальнейшая работа
возможна
 minor-error   (5) ---   незначительная   ошибка   ПО,   данные   не   потеряны,
дальнейшая работа возможна
}

--- блок данных сообщения типа "подтверждение сигнала" (номер сообщения TrapAck
должен соответствовать
номеру сообщения Trap)
TrapAck ::= NULL

END
```

Unformatted.asn

```
Unformatted DEFINITIONS IMPLICIT TAGS ::=
BEGIN

EXPORTS unformattedMessage;

IMPORTS TAGGED,
  sorm-message-unformatted
  FROM Classification

  TelcoList
  FROM Tasks

  Acknowledgement
  FROM Reports

  ConnectionsORIReport
```

```
  FROM ReportsConnectionsORI

  AbonentsORIReport
 FROM ReportsAbonentsORI

   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 ИС ОРИ (других обслуживаемых ОРИ)
 raw-task RawRequestTask --- запрос получения неформатированных данных
}

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

DataStopTypedRequest ::= RawDataType

RawResponse ::= CHOICE {
 data-types-response [0] DataTypesResponse, --- ответ на запрос проверки
наличия вида неформатированных данных
в ИС СОРМ
 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 успешности выполнения команды
DataTypesRequest ::= RawDataType

DataTypesResponse ::= SEQUENCE {
 successful BOOLEAN, --- признак наличия в  ИС СОРМ 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 --- конец временного периода в буфере, по которому
накоплены данные
}

DataStartRequest ::= SEQUENCE {
```

```
 time-from DateAndTime, --- начало временного периода в буфере, с которого
необходимо получить данные
 time-to DateAndTime, --- конец временного периода в буфере, с которого
необходимо получить данные
 raw-type RawDataType --- тип неформатированных данных передачи
}

DataStartResponse ::= BOOLEAN

DataStopRequest ::= NULL
DataStopResponse ::= BOOLEAN

--- типы данных, передаваемых ИС СОРМ
RawDataType ::= ENUMERATED {
 data-reports-ori    (3), --- записи о соединениях абонентов организаторов
распространения информации
 raw-ori (4) --- записи о соединениях абонентов организаторов распространения
информации в бинарном виде
 data-reports-abonents-ori  (5),  --- записи    о принадлежности абонентов
организаторов распространения информации
 raw-abonents-ori  (6), --- записи о принадлежности абонентов организаторов
распространения информации
в бинарном виде
}

RawReport ::= SEQUENCE {
 request-id MessageID, --- идентификатор запроса
 stream-id UTF8String (SIZE (1 .. 256)), --- идентификатор потока в сессии
 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
}

RawDataBlock ::= CHOICE {
 reports-ori [3] ConnectionsORIReport,
 raw-ori [4] RawBytesBlock
 reports-abonents-ori [5] AbonentsORIReport
 raw-abonents-ori [6] RawBytesBlock
}

RawBytesBlock::= SEQUENCE OF RawBytes

RawBytes ::= OCTET STRING (SIZE (1 .. 4096))

RawAcknowledgement ::= Acknowledgement

END".
```

4. В приложении N 8:

а) пункт 1 изложить в следующей редакции:

"1. ПТС ОРИ должны обеспечивать подключение ПУ и обработку поступающих запросов по каналу кпд4 (канал неформатированных данных).";

б) пункт 4 дополнить абзацем следующего содержания:

"ПТС ОРИ и (или) ИС ОРИ должны обеспечивать длительность хранения информации о принадлежности абонентов - 1 сутки со дня регистрации абонента в ИС ОРИ, длительность хранения информации о соединениях абонентов - 3 суток со дня наступления события.";

в) дополнить пунктом 4.1 следующего содержания:

"4.1. Посредством ПТС ОРИ обеспечивается запись в буфер информации о принадлежности абонентов, о соединениях абонентов организаторов распространения информации - файлы в формате "csv" в сжатом виде. Содержимое файлов, содержащих информацию о принадлежности абонентов, генерируется в соответствии с полями, описанными в структуре "AbonentsORIRecord" модуля "ReportsAbonentsORI.asn", содержащейся в приложении N 4 к настоящим требованиям, о соединениях абонентов организаторов распространения информации генерируется в соответствии с полями, описанными в структуре "ConnectionsAAAORIRecord" модуля "ReportsConnectionsORI.asn", содержащейся в приложении N 4 к настоящим требованиям.".