ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р 56845— 2015/ ISO/IEEE 11073-20601:2010

ИНФОРМАТИЗАЦИЯ ЗДОРОВЬЯ

Информационное взаимодействие с персональными медицинскими приборами

Часть 20601

Прикладной профиль. Оптимизированный протокол обмена

ISO/IEEE 11073-20601:2010
Health informatics — Point-of-care medical device communication — Part 20601: Application profile — Optimized exchange protocol (IDT)

Издание официальное

Москва
Стандартинформ
2016

gost.e-ecolog.ru

Предисловие

1 ПОДГОТОВЛЕН Федеральным государственным бюджетным учреждением «Центральный научно-исследовательский институт организации и информатизации здравоохранения Министерства здравоохранения Российской Федерации» (ЦНИИОИЗ Минздрава) и обществом с ограниченной ответственностью «Корпоративные электронные системы» на основе собственного аутентичного перевода на русский язык международного документа, указанного в пункте 4

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 468 «Информатизация здоровья» при ЦНИИОИЗ Минздрава — постоянным представителем ISO ТС 215

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 28 декабря 2015 г. № 2233-ст

4 Настоящий стандарт идентичен международному документу ISO/IEEE11073-20601:2010 «Информатизация здоровья. Информационное взаимодействие с персональными медицинскими приборами. Часть 20601. Прикладной профиль. Оптимизированный протокол обмена» (ISO/IEEE 11073-20601:2010 «Health informatics — Point-of-care medical device communication — Part 20601: Application profile — Optimized exchange protocol»)

Наименование настоящего стандарта изменено относительно наименования указанного международного стандарта для приведения в соответствие с ГОСТ Р 1.5—2004 (пункт 3.5).

При применении настоящего стандарта рекомендуется использовать вместо ссылочных международных стандартов и документов соответствующие им национальные стандарты Российской Федерации, сведения о которых приведены в дополнительном приложении ДА

5 ВВЕДЕН ВПЕРВЫЕ

*Правила применения настоящего стандарта установлены в ГОСТ Р 1.0—2012 (раздел 8). Информация об изменениях к настоящему стандарту публикуется в ежегодном (по состоянию на 1 января текущего года) информационном указателе «Национальные стандарты», а официальный текст изменений и поправок — в ежемесячном информационном указателе «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ближайшем выпуске ежемесячного информационного указателя «Национальные стандарты». Соответствующая информация, уведомление и тексты размещаются также в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет (*gost.e-ecolog.ru

© Стандартинформ, 2016

Настоящий стандарт не может быть полностью или частично воспроизведен, тиражирован и распространен в качестве официального издания без разрешения Федерального агентства по техническому регулированию и метрологии

Содержание

1 Обзор 1
1.1 Область применения 1
1.2 Цель 1
1.3 Контекст 1
2 Нормативные ссылки 4
3 Термины, определения и сокращения 5
3.1 Термины и определения 5
3.2 Сокращения 5
4 Основные принципы 6
5 Введение в персональные медицинские приборы (ИИЭР 11073) 7
5.1 Общие положения 7
5.2 Информационная модель предметной области (DIM) 7
5.3 Сервисная модель 7
5.4 Коммуникационная модель 8
6 DIM персонального медицинского прибора 8
6.1 Общие положения 8
6.2 Использование номенклатуры 9
6.3 Определения классов персональных медицинских объектов 10
6.4 Правила расширения информационной модели 35
7 Сервисная модель персонального медицинского прибора 35
7.1 Общие положения 35
7.2 Сервис ассоциации 36
7.3 Сервисы доступа к объектам 36
7.4 Специальная реализация доступа к объекту сервисов EVENT REPORT для персональных медицинских приборов 36
8 Модель коммуникаций 41
8.1 Общие положения 41
8.2 Контекст системы 41
8.3 Коммуникационные характеристики 42
8.4 Конечные автоматы 44
8.5 Процедура соединения 49
8.6 Неассоциированная процедура 50
8.7 Ассоциированная процедура 50
8.8 Процедура конфигурирования 56
8.9 Процедура Выполнения 58
8.10 Процедура Завершения ассоциации 70
8.11 Кодирование сообщений 71
8.12 Координация времени 71
9 Модель соответствия 74
9.1 Применимость 74
9.2 Спецификация соответствия 74
9.3 Декларации о соответствии реализации (ICS) 75
9.4 Общее соответствие 75
9.5 Дополнения к устройствам/расширения ICS 77
Приложение A (обязательное) Определения языка ASN.1 81
Приложение B (справочное) Пример спецификации шкалы и диапазона 111
Приложение C (справочное) Концепция PM-блока 113
Приложение D (справочное) Типы транспортного профиля 117
Приложение E (обязательное) Таблицы состояний 120
Приложение F (обязательное) Правила кодирования медицинских приборов (MDER) 133
Приложение G (справочное) Закодированные определения типов данных 143
Приложение H (справочное) Примеры 159
Приложение I (обязательное) Коды номенклатуры 165
Приложение J (справочное) История появления и изменения 169
Приложение ДА (справочное) Сведения о соответствии ссылочных международных стандартов и документов национальным стандартам Российской Федерации 171
Библиография 172

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ИНФОРМАТИЗАЦИЯ ЗДОРОВЬЯ

Информационное взаимодействие с персональными медицинскими приборами

Часть 20601

Прикладной профиль. Оптимизированный протокол обмена

Health informatics. Point-of-care medical device communication. Part 20601. Application profile. Optimized exchange protocol

Дата введения — 2016—11—01

1 Обзор

1.1 Область применения

В рамках контекста серии стандартов ИСО/ИИЭР 11073 по взаимодействию приборов настоящий стандарт определяет общую структуру для создания абстрактной модели персональных медицинских данных, доступных в транспортно-независимом синтаксисе передачи, требуемого для установления логических соединений между системами и для обеспечения средств представления и сервисов, необходимых для осуществления задач связи. По мере возможности протокол оптимизируется в соответствии с персональными медицинскими требованиями использования и улучшается за счет общепринятых методов и средств.

1.2 Цель

Настоящий документ уделяет внимание необходимости использования явно определенного, независимого стандарта конвертации информационного профиля в совместимый формат передачи таким образом, чтобы сделать возможным обмен информацией между персональными телемедицинскими приборами и вычислительными системами (например, мобильными телефонами, персональными компьютерами, персональными медицинскими устройствами и декодерами).

1.3 Контекст

Рисунок 1 демонстрирует категории и типичные персональные медицинские приборы. Агенты (например, мониторы артериального давления, весы, педометры) собирают информацию о лице (лицах) и передает эту информацию управляющему устройству-менеджеру (например, на мобильный телефон, персональный компьютер, персональное медицинское устройство) для сбора, отображения и возможной последующей передачи. Управляющее устройство (менеджер) может также направлять данные на удаленные приборы поддержки для проведения дальнейшего анализа. Информация доступна в определенном диапазоне предметных областей, включая приложения об управлении болезнями, здоровьем и фитнесом или программ достойной старости.

Канал связи между агентом и управляющим устройством (менеджером) считается логическим двухточечным соединением. Обычно агент может связываться с единственным управляющим устройством в любой момент времени. Управляющее устройство может связываться одновременно с несколькими агентами, используя отдельные двухточечные соединения.

Графическая пунктирная накладка показывает главное направление деятельности Рабочей группы по персональным медицинским приборам ИИЭР 11073™. Первостепенный интерес представляет интерфейс и обмен данными между агентами и управляющим устройством. Однако данный интерфейс

Издание официальное

невозможно создать, игнорируя остальную часть пространства решений. Принимая во внимание всю систему в целом можно обеспечить целесообразную передачу данных от агентов к удаленным службам поддержки. Данный подход может включать конвертацию формата данных, протоколов обмена, и транспортных протоколов на различных интерфейсах. Большая часть мер по стандартизации находится вне области применения Рабочей группы по персональным медицинским приборам; однако объединение мер по стандартизации позволяет осуществлять бесперебойную передачу данных через все множество систем.

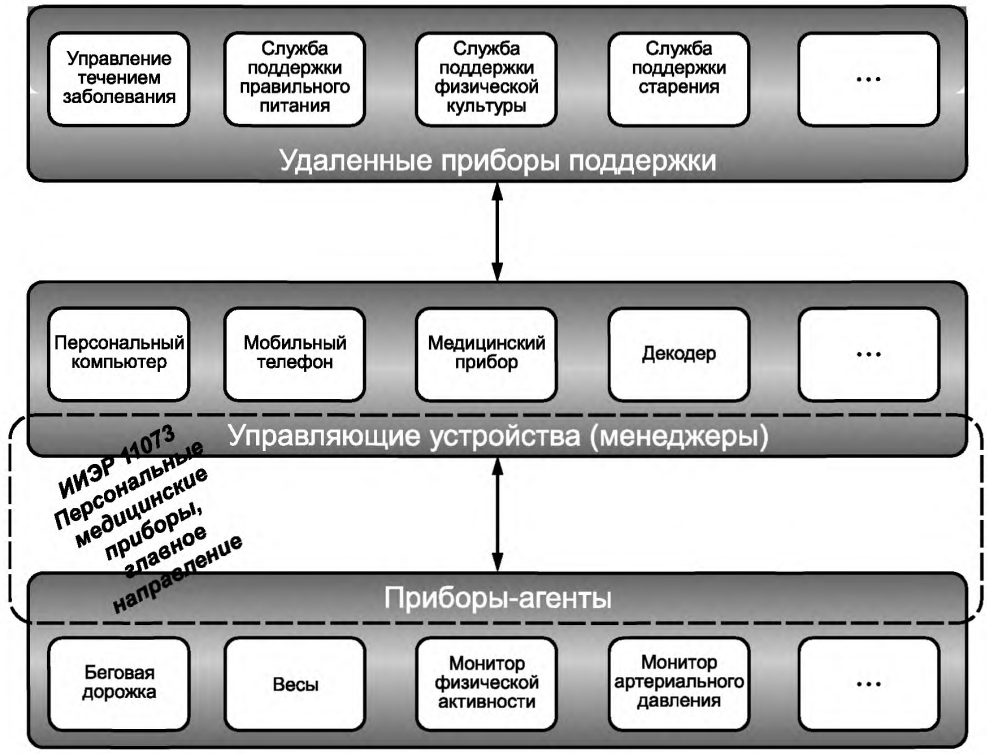


Рисунок 1 — Общий контекст работы

Рисунок 2 демонстрирует иерархическое представление архитектуры агента или управляющего устройства с указанием соответствующих стандартов. Уровни приложений в основном не относятся к какому-либо конкретному транспортному протоколу. Там, где это необходимо, настоящий стандарт определяет допущения, которые требуют прямой поддержки транспортным протоколом или некоторого «промежуточного» уровня над ним. Данный подход позволяет оказывать поддержку различным видам транспортных протоколов. Определение транспортного протокола лежит вне рамок области применения настоящего стандарта и рабочей группы.

Выше транспортного уровня находится Оптимизированный Протокол Обмена (описанный в настоящем стандарте). Данный протокол состоит из двух аспектов: сервисы уровня приложений и определение протокола обмена данными между агентами и управляющими устройствами. Сервисы уровня приложений предоставляют протокол для управления соединением и надежной передачей действий и данных между агентом и управляющим устройством. Протокол обмена данными определяет команды, информацию о конфигурациях агента, формат данных и общий протокол. Оптимизированный Протокол Обмена обеспечивает основу для поддержки агента любого вида. В случае определенного типа прибора читателю необходимо обратиться к техническим характеристикам прибора для этого агента, чтобы понять свойства данного прибора и условия его применения в соответствии с настоящим стандартом. Специализация прибора указывает на то, какие аспекты настоящего стандарта следует охватить и где найти дополнительную информацию по применению данного прибора.

Над протоколом обмена находятся специализации, которые описывают специфические детали относительно конкретного агента (например, монитор артериального давления, весы, педометр). Специализации дают подробную информацию о том, как данные агенты работают и действуют, представленную в виде детального описания для создания конкретного типа агента. А также они дают ссылки на соответствующий стандарт для получения дополнительной информации. Номера стандартов по специализациям приборов варьируются от ИИЭР 11073-10401 до ИИЭР 11073-10499 включительно. При приведении ссылок на стандарты используется следующая схема: ИИЭР 11073-104zz, где zz может быть любым числом от 01 до 99 включительно.

Технический отчет ИСО/ИИЭР Р11073-090103 [8] описывает личное пространство персональных медицинских приборов в целом с последующим пояснением базовых вариантов использования и моделей применения.



Рисунок 2 — Общая схема настоящего стандарта

Специализации персонального медицинского прибора не создаются независимо от других стандартов. Существует определенное количество стандартов, созданных для клинических сред, на которые опираются эти стандарты. Рисунок 3 демонстрирует связь с остальными документами ИИЭР 11073. Существует два типа таких связей:

- использование базовых идей и/или фрагментов из других документов (пунктирные линии);
- эффективное использование информации из другого документа и введение нового текста в документ для поддержки настоящего стандарта (сплошные линии).

Настоящий стандарт вносит информацию из ИСО/ИИЭР 11073-10201:2004 [13] и ИСО/ИИЭР 11073-20101:2004 [14] в качестве нормативных приложений. При наличии разногласий между данными стандартами, настоящий стандарт считается приоритетным. В связи с повторным использованием структур из рассматриваемых стандартов, некоторые названия могут быть более ориентированными на медицинские учреждения (например, Система медицинского прибора (MDS) вместо системы персонального медицинского прибора); однако, для сохранения согласованности, были сохранены традиционные названия.

Настоящий стандарт копирует соответствующие параграфы ИСО/ИИЭР 11073-10101 [12] и включает новые номенклатурные коды.

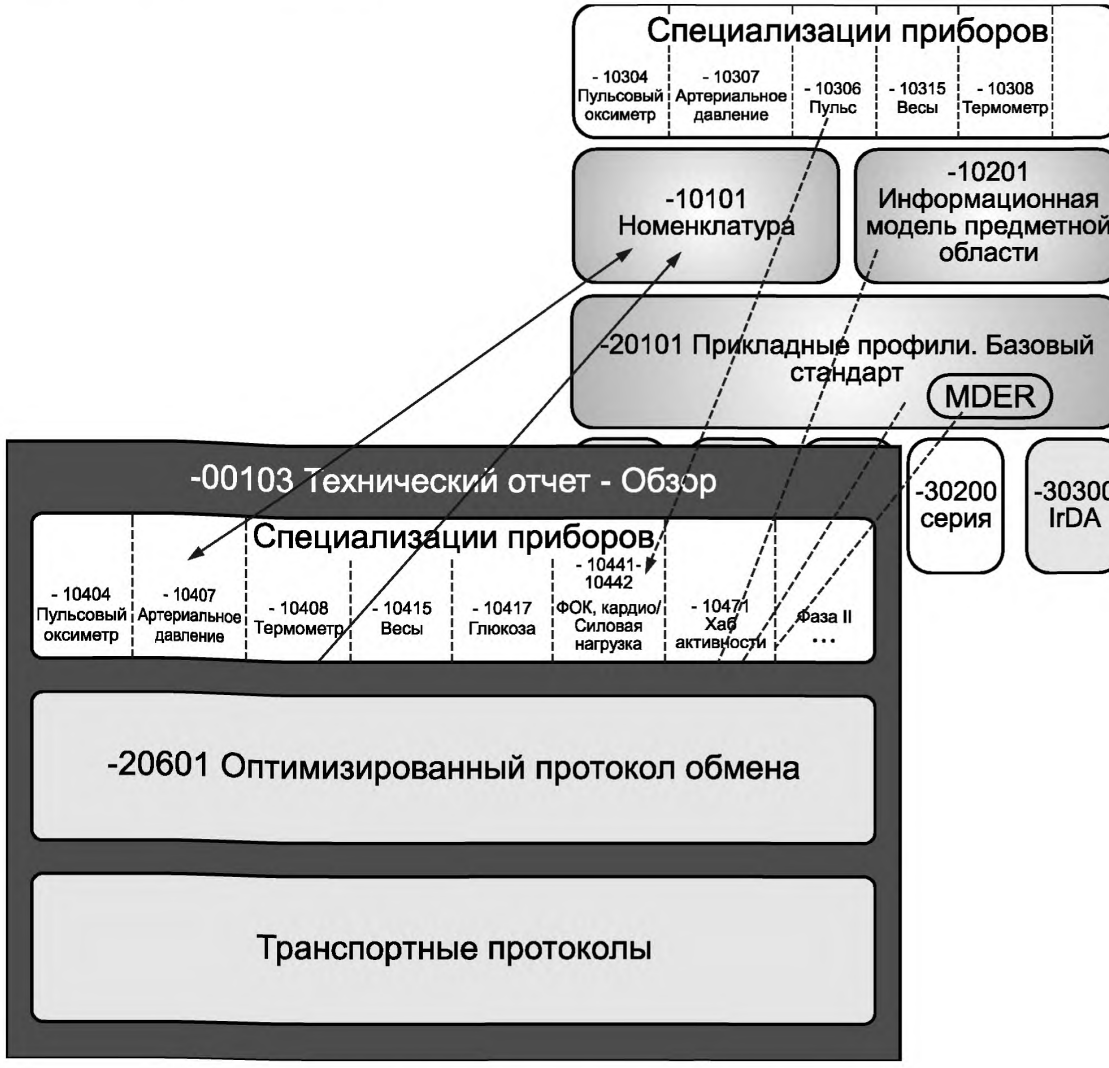


Рисунок 3 — Связь с другими документами ИИЭР 11073

2 Нормативные ссылки

Для применения настоящего стандарта необходимы следующие ссылочные документы. Для датированных ссылок применяют только указанное издание ссылочного документа, для недатированных ссылок применяют последнее издание ссылочного документа (включая все его изменения).

ИИЭР 802-2001 Стандарт ИИЭР для локальных вычислительных сетей и сетей мегаполисов. Обзор и архитектура (ИИЭР Std 802®-2001, ИИЭР Standard for Local and Metropolitan Area Networks: Overview and Architecture[1)])

Рекомендации сектора электросвязи МСЭ X.667 (сентябрь 2004 г.) Информационная технология. Взаимосвязь открытых систем. Процедуры для работы органов регистрации семиуровневой сетевой

[1)] Публикации ИИЭР доступны в Институте инженеров по электротехнике и радиоэлектронике, 45 Hoes Lane, Piscataway, NJ 08854, USA (gost.e-ecolog.ru).

модели ВОС: Генерация и регистрация универсально уникальных идентификаторов (UUID) и их использование в качестве компонентов идентификатора объектов языка ASN.1 (ITU-T Rec. X.667 [Sept. 2004], Information technology — Open Systems Interconnection — Procedures for the operation of OSI Registration Authorities: Generation and registration of universally unique identifiers (UUIDs) and their use as ASN.1 object identifier components[2])

3 Термины, определения и сокращения

3.1 Термины и определения

В настоящем стандарте применяются следующие термины и определения. Для получения информации о терминах, не указанных в данном разделе, см. [6].

3.1.1 **агент** (agent): Узел, который собирает и передает персональные медицинские данные ассоциированному управляющему устройству (менеджеру).

3.1.2 **вычислительное устройство** (compute engine): См. **управляющее устройство** (manager).

3.1.3 **подтвержден** (confirmed): Механизм службы уведомления о завершении на уровне приложения. Для сервисов EVENT REPORT (при передаче данных) подтверждение позволяет агенту понять, когда управляющее устройство «приняло на себя ответственность» за данную часть информации, после чего агент может ее удалить. Для сервисов ACTION, GET и SET (при управлении) подтверждение позволяет управляющему устройству понять, когда агент «завершает» запрашиваемую операцию.

3.1.4 **прибор** (device): Физический прибор, выполняющий роль либо агента, либо управляющего устройства.

3.1.5 **идентификатор** (handle): Беззнаковое 16-битное число, являющееся уникальным, и идентифицирующее один из объектов-экземпляров внутри агента.

3.1.6 **управляющее устройство**, **менеджер** (manager): Узел, получающий данные от одной или нескольких агентских систем. Примерами менеджеров могут служить мобильные телефоны, медицинское устройство, декодер или компьютерная система.

3.1.7 **персональный медицинский прибор** (personal health device): Прибор, используемый для личного медицинского применения.

3.1.8 **персональный удаленный медицинский прибор** (personal telehealth device): См. **персональный медицинский прибор**.

3.2 Сокращения

В дополнении к сокращениям стандарта ИИЭР 1073 в настоящем стандарте используются следующие сокращения:

ASCII — американский стандартный код для обмена информацией[3];
ASN.1 — язык asn.1 для описания абстрактного синтаксиса;
APDU — блок данных протокола прикладного уровня;
AVA — установление значения атрибута;
BER — правила бинарного кодирования;
DIM — информационная модель предметной области;
EUI-64 — расширенный уникальный идентификатор (64 бит);
GMDN — всемирная номенклатура медицинских изделий;
ICS — заявление о соответствии применения;
ID — идентификатор/идентификационная информация;
LSB — младший двоичный бит;
MDER — правила кодирования медицинских приборов;
MDNF — цифровой формат медицинского прибора;
MDS — система медицинского прибора;
MOC — класс медицинского объекта;

---

[2] Публикации ITU-T доступны в Международном союзе электросвязи (Place des Nations, CH-1211, Geneva 20, Switzerland/Suisse (gost.e-ecolog.ru

[3] В настоящем стандарте термин ASCII используется в значении набора знаков в соответствии с ISO/IEC 646 (1991) [B7].

MSB — старший двоичный бит;
NaN — не число;
NBO — порядок передачи байтов;
NRes — не при этом разрешении экрана;
NTP — временной сетевой протокол;
OID — идентификатор объекта;
OUI — идентификатор, уникальный в пределах организации;
PDU — блок данных протокола;
PER — правила уплотненного кодирования (ASN.1);
POC — объект и класс информационной модели предметной области персональных медицинских приборов;
RC_{assoc} — счетчик повторений в процедуре ассоциации;
RTC — часы реального времени;
RT-SA — массив показаний, снятых в режиме реального времени;
SNTP — простой временной сетевой протокол;
TCP — управляющий протокол передачи;
TO_{assoc} — время работы процедуру ассоциации;
TO_{ca} — время работы сервиса подтвержденного действия;
$TO_{cer-mds}$ — время работы сервиса подтвержденного отчета о событии для объекта системы медицинского прибора;
$TO_{cer-pms}$ — время работы сервиса подтвержденного отчета о событии для объекта РМ-блока;
$TO_{cer-scan}$ — время работы сервиса подтвержденного отчета о событии для объекта сканера;
$TO_{clr-pms}$ — время работы сервиса подтвержденного события для очистки объекта РМ-блока;
TO_{config} — время выполнения процедуры конфигурации;
TO_{cs} — время работы сервиса подтвержденной установки;
TO_{get} — время работы сервиса получения (get);
$TO_{release}$ — время работы процедуры разъединения ассоциации;
TO_{sp-mds} — специальный период времени между сервисами для объектов систем медицинского прибора;
TO_{sp-pms} — специальный период времени передачи сегмента для объекта РМ-сегмента;
UTC — универсальное время;
UUID — универсальный уникальный идентификатор;
USB — универсальная последовательная шина;
XER — правила кодировки расширяемого языка разметки (XML).

4 Основные принципы

Настоящий стандарт и другие стандарты о персональных медицинских приборах включены в широкий контекст набора стандартов ИСО/ИИЭР 11073. Полный пакет программ позволяет агентам осуществлять взаимную связь и работу с управляющими устройствами и компьютеризованными информационными медицинскими системами.

Коммуникационный профиль, определенный в настоящем стандарте, учитывает следующие специфические требования медицинских агентов и управляющих устройств, обычно используемых за пределами здания больницы, например, переносное оборудование или оборудование, используемое пациентами в домашних условиях:

- медицинские агенты обычно имеют очень ограниченные вычислительные возможности;

- медицинские агенты обычно обладают фиксированной конфигурацией, и используются с одним управляющим устройством;

- медицинские агенты являются переносными приборами, постоянно работающими от аккумулятора и использующими беспроводной тракт. Следовательно, эффективность энергоиспользования протокола является важным аспектом;

- медицинские агенты часто не являются постоянно активными. Например, весы могут предоставлять данные только один или два раза в день. Для минимальных эксплуатационных расходов для подобных приборов необходима процедура эффективной связи;

- медицинские управляющие устройства обычно обладают большей производительностью, памятью и объемом памяти, таким образом, протокол намеренно устанавливает больше нагрузки на управляющие устройства;

- медицинские агенты и управляющие устройства передают информацию, которая может быть полезна для медицинского персонала. В этом случае, качество данных может рассматриваться как обладающее медицинской ценностью, даже если они получены в медицинской среде или в среде удаленного контроля.

Комплект стандартов ИСО/ИИЭР 11073 основывается на парадигме управления объектно-ориентированных систем. Данные (измерения, состояние и т. д.) моделируются в форме информационных объектов, доступ к которым можно получить при помощи протокола обслуживания доступа к объектам.

Для удовлетворения требованиям персональных медицинских приборов, в настоящем стандарте дается определение специализированному прикладному профилю. Данный профиль использует идеи серии стандартов ИСО/ИИЭР 11073 и успешный отраслевой опыт для определения оптимизированного профиля связи для данной предметной области:

- там, где это возможно, профиль связи не конкретизируется под какой-либо транспортный протокол;

- информационная модель профиля связи строится на информационной модели предметной области (DIM) ИСО/ИИЭР 11073 и включает выбор оптимальных параметров, где это необходимо;

- оптимизированный протокол связи определяется для уменьшения размера сообщения, времени создания пакета и издержек при синтаксическом разборе. Это представляется возможным благодаря меньшей сложности приборов в области персональной медицины;

- требуемые определения для применения протокола включены в настоящий стандарт, в отличие от ссылочных документов.

Данный подход способствует более простому применению настоящего стандарта. В случае расхождений между нормативными включениями и ссылочным документом, считать приоритетным настоящий стандарт.

Там, где это возможно, редакция настоящего стандарта полностью совместима, по крайней мере, с двумя последними редакциями.

Примечание — Ожидается, что любые добавления в информационную модель предметной области или другие соответствующие части серии стандартов ИСО/ИИЭР 11073 будут приняты и рассмотрены в последующих редакциях этих стандартов.

5 Введение в персональные медицинские приборы (ИИЭР 11073)

5.1 Общие положения

Общая системная модель ИСО/ИИЭР 11073 подразделяется на три основных компонента: информационная модель предметной области, сервисная модель и коммуникационная модель. Эти три модели работают вместе для представления данных, определения доступа к данным и методологий команд, а также для передачи данных от агента к менеджеру. По причине тесной связи между моделями информационная модель предметной области, сервисная модель и коммуникационная модель кратко представлены в 5.2, 5.3 и 5.4, соответственно, и более подробно описаны в разделах 6, 7 и 8, соответственно.

5.2 Информационная модель предметной области (DIM)

Информационная модель предметной области (DIM), детально описанная в 6, характеризует информацию, поступающую от агента как набор объектов. Каждый объект обладает одним или несколькими атрибутами. Атрибуты описывают данные об измерениях, которые передаются менеджеру, а также элементы, которые контролируют поведение и отправляют отчет о статусе агента.

5.3 Сервисная модель

Сервисная модель, детально описанная в 7, обеспечивает элементарные процедуры доступа к данным, посылаемые между агентом и менеджером для обмена данными из DIM. Эти элементарные процедуры включают в себя такие команды как Get (Получить), Set (Установить), Action (Действие) и Event Report (Отчет о событии).

5.4 Коммуникационная модель

Коммуникационная модель, детально описанная в 8, поддерживает топологию одного или нескольких агентов, взаимодействующих через двухточечные соединения с одним менеджером. Для каждого двухточечного соединения, поведение динамической системы определяется конечным автоматом соединения. Конечный автомат соединения определяет состояния и подсостояния, через которые проходит пара менеджер и агент, включая состояния, имеющие отношение к соединению, взаимосвязи и работе. Коммуникационная модель также детально определяет входные и выходные данные, условия возникновения ошибок для соответствующих состояний, включая различные рабочие процедуры для передачи измеряемых данных. Коммуникационная модель включает в себя также предположения относительно поведения нижних коммуникационных уровней.

Другой функцией коммуникационной модели является преобразование абстрактных данных моделирования (абстрактный синтаксис), используемых в модели DIM, в синтаксис передаваемых данных, например, в двоичные сообщения при помощи правил кодирования медицинских приборов (MDER), посылаемых коммуникационной моделью.

6 DIM персонального медицинского прибора

6.1 Общие положения

Персональные медицинские приборы, в рамках настоящего стандарта, определяются посредством объектно-ориентированной модели. Данная модель DIM определяет несколько классов для моделирования агента. Модель описывает прибор-агент, как набор объектов, которые представляют источники данных, как элементы, которые менеджер может использовать для контроля поведения агента, и как механизм, который использует агент для отчета об обновлениях в статусе представления агента. Объекты прибора-агента обладают атрибутами, которые отображают информацию и статус объекта.

Приборы-менеджеры связываются с объектами прибора-агента посредством использования строго определенных методов, таких как GET и SET, и определенных в каждом подразделе, описывающим объект. Такая информация, как измерения, пересылается от объектов данных агента к прибору-менеджеру при помощи отчетов о событиях.

Информационная модель предметной области персональных медицинских приборов является объектно-ориентированной моделью, определяющей объекты данных агентов, включая их атрибуты и методы. Использование объектно-ориентированной информационной модели оказывает поддержку в следующих случаях:

- для отделения спецификации от применения посредством принципа инкапсуляции;
- для поддержки эволюции посредством принципа наследования;
- для поддержки соответствия с предыдущими версиями посредством принципа полиморфизма.

Объекты, происходящие из классов, определенных в информационной модели, представляют все данные, которые система агента может передавать системе менеджера при помощи прикладного протокола, определенного настоящим стандартом. Такие данные моделируются в форме объектных атрибутов. Более того, информационная модель определяет сервисы доступа к специфическим данным в форме методов, использованных для обмена данными между системами агента и менеджера. Эти сервисы моделируют сообщения прикладного протокола (элементарные процедуры доступа к данным) определенного в настоящем стандарте.

Объекты определяют структуру и возможности системы агента. Система менеджера входит в эти объекты для сбора данных и/или для контроля системы агента. Настоящий стандарт не определяет информационную модель системы менеджера.

Информационная модель — это иерархическая модель, которая определяет логическую структуру и возможности персонального медицинского прибора. На верхнем уровне объект системы медицинского прибора (MDS) представляет характеристики и сервисы собственно прибора, независимо от возможностей передачи медицинских данных. Возможности объекта MDS включают в себя атрибуты для идентификации прибора и последующих технических пояснительных надписей и данных о статусе. Прикладные данные (например, медицинские или измерительные данные), предоставляемые персональным медицинским прибором моделируются в форме объектов дополнительной информации, которые логически содержатся в объекте MDS. Набор атрибутов объектов вместе с данным отношением включения описывает конфигурацию и, в этой связи, возможности персонального медицинского прибора.

Необходимо помнить, что в то время как определения в настоящем стандарте применяют объектную ориентацию для определения информационной модели, данная практика не подразумевает использование объектно-ориентированных технологий (например, объектно-ориентированные языки программирования) для применения настоящего стандарта в определенных приборах. Модель используется для определения структуры данных и методов доступа (сообщения протокола) надежным способом. Соответствие этим определениям находится только на уровне сообщения протокола связи. В частности, определения в настоящем стандарте оптимизированы, чтобы обеспечить очень простые применения агентов (например, посредством использования предустановленных шаблонов передачи). Более того, применение прибора-менеджера обуславливает выбор программной структуры, использующей информационные объекты, в отличие от других альтернативных структур.

Настоящий стандарт использует информационные классы и объекты, которые были определены в ИСО/ИИЭР 11073-10201:2004 [13], при этом адаптируя их под область коммуникаций персональных медицинских приборов следующим образом:

- определение обязательных, необязательных и условных атрибутов могут отличаться;
- могут не приводиться определения сервисов дополнительного объекта;
- могут приводиться определения дополнительных атрибутов;
- некоторые свойства оригинальной модели могут не использоваться.

6.2 Использование номенклатуры

Ключевым аспектом модели DIM является тот факт, что классы и атрибуты классов обеспечиваются при помощи номенклатурных кодов, перечисленных в ИСО/ИИЭР 11073-10101 [12]. Используя согласованную номенклатуру, обеспечивается межоперационная совместимость, так как все применения обладают одним и тем же семантическим значением для цифровых кодов. Использование номенклатурных кодов помогает также при международном применении, так как использование строк снижено.

Номенклатура ИСО/ИИЭР 11073 определяется как набор контекстно-зависимых разделов. Номенклатурный код в каждом контекстно-зависимом разделе определяется 16-битным кодом, который поддерживает до 65 536 независимых терминов в разделе. На разделы ссылаются 16-битным кодом раздела. Если раздел номенклатурного кода определяются посредством контекста, тогда становится возможным использовать только 16-битный код. Если контекст не определен или требуется контекстно-независимый код, тогда эта ситуация задается 32-битным кодом, построенным из 16-битного кода раздела вместе с 16-битным кодом термина. Таблица 1 демонстрирует разделы, которые определены в настоящем стандарте и/или ИСО/ИИЭР 11073-10101 [12].

Коды термина от 0xF000 до 0xFFFF в каждом разделе в номенклатуре предусмотрены для частных номенклатурных кодов.

Для каждого номенклатурного термина ИСО/ИИЭР 11073-10101 [12] определяет систематические названия, которые объясняют этот термин, значение уникального кода, и ссылочный идентификатор (ID). Ссылочный ID указываются в форме MDC_XXX_YYY (MDC указывает на «коммуникацию медицинского прибора»). В тексте настоящего стандарта номенклатурные термины и номенклатурные коды указываются в ссылочном ID.

Таблица 1 — Разделы в номенклатуре

Номер раздела	Номенклатурная категория
1	Объектно-ориентированная (ОО)
2	Дистанционное управление и сбор данных (SCADA)
3	События
4	Размеры (единицы измерения)
5	Виртуальные атрибуты
6	Группы параметров
7	Участки (тела)
8	Инфраструктура

Окончание таблицы 1

Номер раздела	Номенклатурная категория
9—127	Резерв
128	Менеджмент заболеваний персонального медицинского прибора
129	Здоровье и фитнес персонального медицинского прибора
130	Достойная старость персонального медицинского прибора
131—254	Резерв
255	Коды возврата
256	Внешние номенклатурные ссылки
257—1023	Резерв
1024	Частная
1025—65 535	Резерв

6.3 Определения классов персональных медицинских объектов

6.3.1 Общие положения

Схема на рисунке 4 использует Унифицированный язык моделирования (UML) для представления информационных объектов персонального медицинского агента вместе с отношениями классов. Верхний объект представляет информацию и статус системы MDS (см. 6.3.2). Существует ноль или более числовых, массивов показаний, снятых в режиме реального времени (RT-SA), перечислений, сканеров или других объектов PM-блока, ассоциированных с объектом системы MDS. Существует ноль или больше PM-сегментов, которые содержат постоянные метрики, ассоциированные с PM-блоком. Числовые объекты, объекты RT-SA и объекты перечисления происходят из базового класса общей метрики, который содержит общие и разделяемые атрибуты (см. 6.3.3). В общем случае, числовые объекты представляют эпизодические измерения (см. 6.3.4), объекты RT-SA представляют непрерывные показания или биосигналы (см. 6.3.5), объекты перечисления представляют аннотации событий (см. 6.3.6), а PM-блоки (см. 6.3.7) вместе с PM-сегментами (см. 6.3.8) предоставляют постоянные механизмы хранения метрики, доступ к которой позже получает менеджер. Кроме того, объект сканера (подробно определенный в 6.3.9) облегчает сообщения о передаче данных, инициированных агентом.

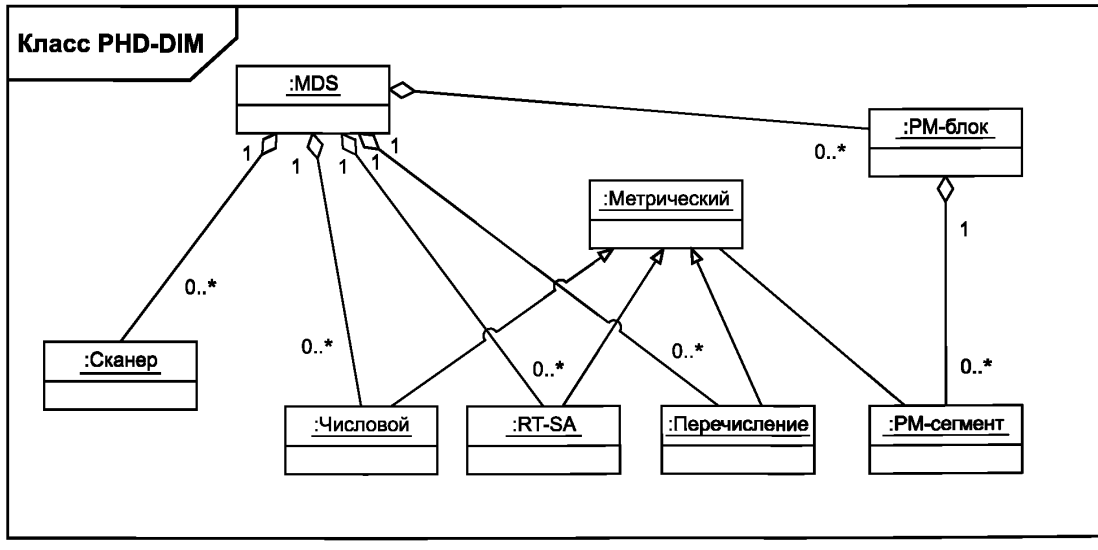


Рисунок 4 — Персональный медицинский прибор. DIM

Пункты с 6.3.2 по 6.3.9 описывают классы персонального медицинского прибора модели DIM. Каждый подпункт использует следующий формат:

- номенклатурный код, используемый для идентификации класса. Данный код используется в течение конфигурации для сообщения о классе для каждого объекта и позволяет менеджеру узнавать, является ли указанный объект числовым, RT-SA, или любого другого класса;
- атрибуты, определяемые классом;
- доступные методы;
- потенциальные события, генерируемые объектами инстанциируются от классов;
- доступные сервисы, такие как получение или установка атрибутов.

Каждый тип атрибутных данных определяется при помощи абстрактной синтаксической нотации версии 1 (ASN.1). Определения ASN.1 для всех типов данных и форматов обмена находятся в приложении А.

Атрибуты для каждого класса определены в таблицах, которые представляют название атрибута, его номенклатурную ссылку ID, тип, описание атрибута, и его классификатор. Классификаторы со значением О — необязательный атрибут, М — обязательный атрибут и С — условный атрибут, которые зависят от условий, указанных в столбце «Комментарий». Условные атрибуты могут реализовываться в отношении агента. Обязательные атрибуты должны реализовываться агентом. Условные атрибуты должны реализовываться при действии условия, а также в других случаях.

Номенклатурный код классов объектов (например, числовой, RT-SA) пересылается менеджеру во время конфигурирования для создания зеркального представления объекта. Каждый объект обладает атрибутом Handle (дескриптор), который используется для идентификации объекта для операций (от и к объекту), а также другими атрибутами для представления и передачи информации на физический прибор и его источники данных. Доступ к атрибутам и возможность внесения изменений предоставляется посредством таких методов, как GET (ПОЛУЧИТЬ) и SET (УСТАНОВИТЬ). Данные передаются при помощи метода EVENTs (СОБЫТИЯ).

6.3.2 Класс MDS

6.3.2.1 Общие положения

Каждый персональный медицинский агент определяется объектно-ориентированной моделью, указанной на рисунке 4. Объект верхнего уровня каждого агента инстанцируются от класса MDS. Каждый агент обладает одним объектом MDS. MDS представляет идентификационную информацию и статус агента посредством его атрибутов.

6.3.2.2 Идентификация класса MDS

Номенклатурный код для обозначения класса MDS:MDC_MOC_VMS_MDS_SIMP.

6.3.2.3 Атрибуты классов MDS

Таблица 2 определяет набор атрибутов MDS, поддерживаемых для связи персональных медицинских агентов. Объект MDS должен поддерживать все обязательные атрибуты, но может иметь подмножества условных и необязательных атрибутов.

Таблица 2 — Атрибуты MDS

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластор
Handle	MDC_ATTR_ID_ HANDLE	Handle	Атрибут Handle представляет ссылочный ID для данного объекта. Значение атрибута MDS Handle должно составлять 0	M
System-Type	MDC_ATTR_ SYS_TYPE	System-Type	Данный атрибут определяет тип агента в соответствии с номенклатурой	C
			(например, весы). Значения должны быть получены из ИСО/ИИЭР 11073-10101 [12], раздела nom-part-object и подраздела MD-Gen (Медицинский Прибор — Общий). Должен присутствовать либо Данный атрибут либо System-Type-Spec-List. Данный атрибут должен оставаться неизменным после одобрения конфигурации	

Продолжение таблицы 2

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кла-тор
System-Model	MDC_ATTR_ID_ MODEL	System-Model	Данный атрибут определяет производителя и номер модели прибора-агента. Данный атрибут должен оставаться неизменным после одобрения конфигурации	M
System-Id	MDC_ATTR_ SYS_ID	OCTET STRING	Данный атрибут принадлежит к ИИЭР EUI-64, который состоит из 24-битного идентификатора, уникального в пределах организации (OUI), сопровождаемого 40-битным ID, определенным производителем. Значение OUI должно быть предписано Регистрационным органом ИИЭР gost.e-ecolog.ru ИИЭР.org/regauth/index.html) и должно использоваться в соответствии со стандартом ИИЭР Std 802-2001.6 Данный атрибут должен оставаться неизменным после одобрения конфигурации	M
Dev-Configuration-Id	MDC_ATTR_ DEV_CONFIG_ID	ConfigId	Данный атрибут определяет идентификацию конфигурации прибора-агента. Данный Dev-Configuration-Id является статичным во время работы ассоциации; обычно он меняется в течение процедуры ассоциации. Менеджер может получить данный атрибут во время работы. Если данный атрибут встает в очередь до того, как агент и менеджер одобрят конфигурацию, агент должен вернуть ID конфигурации, предлагаемый в этот момент времени. Для более подробной информации по данному атрибуту, смотреть пункт 8.7.6. Данный атрибут должен оставаться неизменным после одобрения конфигурации	M
Attribute-Value-Map	MDC_ATTR_ ATTRIBUTE_ VAL_MAP	AttrValMap	Данный атрибут определяет атрибуты, которые сообщаются в фиксированном формате сообщений об обновлении данных (см. 7.4.5 для более подробной информации). Использование данного атрибута обязательно, если прибор-агент использует сообщения с значениями фиксированного формата для сообщения динамических данных для объекта	C
Production-Specification	MDC_ATTR_ID_ PROD_SPECN	Production-Spec	Данный атрибут определяет ревизии компонента, серийные номера и т.д. в формате производителя. Данный атрибут должен оставаться неизменным после одобрения конфигурации	O
Mds-Time-Info	MDC_ATTR_ MDS_TIME_ INFO	MdsTimeInfo	Данный атрибут определяет способности обработки времени и статус MDS. Использование этого атрибута требуется при поддержке синхронизации или настройки времени	C
Date-and-Time	MDC_ATTR_ TIME_ABS	AbsoluteTime	Данный атрибут определяет дату и время агента с точностью в 1/100 секунды. Для более подробной информации о данном атрибуте, см. 8.12. Если агент сообщает об абсолютном значении времени (AbsoluteTime) в любом из его сообщений, он должен сообщить его текущее значение абсолютного временив данном атрибуте	C
Relative-Time	MDC_ATTR_ TIME_REL	RelativeTime	Если агент сообщает о Относительном времени (RelativeTime) в своем сообщении, он должен сообщить текущее значение Относительного времени (RelativeTime) в данном атрибуте	C

Продолжение таблицы 2

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластер
HiRes-Relative-Time	MDC_ATTR_TIME_REL_HI_RES	HighResRelativeTime	Если агент сообщает о HighResRelativeTime в своем сообщении, он должен сообщить текущее значение HighResRelativeTime в данном атрибуте	C
Date-and-Time-Adjustment	MDC_ATTR_TIME_ABS_ADJUST	AbsoluteTimeAdjust	Данный атрибут сообщает о любых настройках даты и времени, которые выполняются в связи с изменением времени человеком или в связи с такими событиями, как переход на летнее время. Используется только в отчетах о событиях. Если запрашивается при помощи команды MDS объекта «Get», данное значение либо не должно присутствовать, либо равно нулю. Если агент осуществляет настройку времени и даты, данный атрибут используется в отчете о событиях для сообщения о таких изменениях	C
Power-Status	MDC_ATTR_POWER_STAT	PowerStatus	Данный атрибут сообщает, осуществляется ли потребление энергии от батареи или от электросетей, а также статус заряда	O
Battery-Level	MDC_ATTR_VAL_BATT_CHARGE	INT-U16	Данный атрибут сообщает процентное соотношение оставшейся емкости батареи, которое не определяется, если значение > 100	O
Remaining-Battery-Time	MDC_ATTR_TIME_BATT_REMAIN	BatMeasure	Данный атрибут представляет прогнозируемый объем оставшегося рабочего времени аккумуляторов. В блок BatMeasure устанавливается одно из значений MDC_DIM_MIN, MDC_DIM_HR или MDC_DIM_DAY для указания минут, часов или дней соответственно	O
Reg-Cert-Data-List	MDC_ATTR_REG_CERT_DATA_LIST	RegCert-DataList	Данный атрибут перечисляет различные регуляторные и/или сертификационные пункты совместимости, на которые ссылается агент. Реализация Заявки о соответствии (см. пункт 9) имеет преимущественную силу над данным атрибутом и являются документами, имеющими юридическую силу. Данный атрибут должен оставаться неизменным после одобрения конфигурации	O
System-Type-Spec-List	MDC_ATTR_SYS_TYPE_SPEC_LIST	TypeVerList	Данный атрибут сообщает тип(-ы) агента, в соответствии с номенклатурой (например, весы). Значения должны происходить из ИСО/ИИЭР 11073-10101 [12], раздела nom-part-infrastruct, подраздела DEVspec, и ссылка на специализации ИСО/ИИЭР 11073-104zz. Если агент не следует ни одной из специализации, перечень должен остаться пустым. Данный перечень должен также содержать редакции специализации. Должен присутствовать либо данный атрибут, либо Тип Системы (System-Type). Если агент мультифункциональный, то должен присутствовать данный атрибут. Данный атрибут должен оставаться неизменным после одобрения конфигурации	C

Окончание таблицы 2

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластер
Confirm-Timeout	MDC_ATTR_CONFIRM_TIMEOUT	RelativeTime	Данный атрибут информационного таймаута определяет минимальное время, в течение которого агент должен ожидать сообщения ответа от менеджера после выдачи сообщений запроса Подтвержденного Отчета о Событиях до тайм-аута и перевода в неассоциированное состояние. Это информационный атрибут в пользу менеджера. Если поставляется данный атрибут, он должен соответствовать текущему значению тайм-аута, который используется агентом для Подтвержденного Отчета о Событиях, выдаваемого объектом MDS. Данный атрибут является информационным для менеджера в том смысле, что менеджер не использует данный атрибут в текущей реализации протокола (т.е. менеджер не производит тайм-аут на Подтвержденный Отчет о Событиях, сгенерированный агентом). Однако менеджер может пожелать использовать данную информацию для назначения приоритетов обработки агента «короткого» тайм-аута над агентом «долгого» тайм-аута	O

Типы атрибутивных данных определены в приложении А

6.3.2.4 Методы объектов MDS

Таблица 3 определяет методы (действия), доступные для объектов системы MDS. Данные методы запрашиваются при помощи сервиса ACTION. В таблице 3 столбец Метод/Действие определяет наименование метода. Столбец «Режим» определяет, был ли метод запущен как неподтвержденное действие (т.е. roiv-cmip-action из А.10.2) или подтвержденное действие (т.е. roiv-cmip-confirmed-action). Столбец «Тип Действия» определяет ID номенклатуры для использования в поле «тип действия» запроса и ответа действия (см. А.10.6). Столбик action-info-args определяет структуру ассоциированных данных для использования в сообщениях действий для поля запроса action-info-args. Столбец «Результат action-info-args» определяет структуру, используемую для ответа на action-info-args.

Таблица 3 — Методы объектов MDS

Метод/действие	Режим	Тип действия	action-info-args	Результат action-info-args
MDS-Data-Request	подтвержденный	MDC_ACT_DATA_REQUEST	Запрос данных (DataRequest)	Ответ данных (DataRequest)
Set-Time	подтвержденный	MDC_ACT_SET_TIME	Вызов установки времени (SetTimeInvoke)	нет

Запрос данных MDS (MDS-Data-Request).

Данный метод позволяет системе менеджера активировать или дезактивировать передачу измерительных данных от агента (см. 8.9.3.3.3 для подробного описания).

Установка времени (Set-Time).

Данный метод позволяет системе менеджера устанавливать часы реального времени (RTC) с абсолютным временем. Агент указывает, корректна ли команда Установки времени при помощи mds-time-capab-set-clock bit в атрибуте Mds-Time-Info (см. таблицу 2).

6.3.2.5 События объектов MDS

Таблица 4 определяет потенциальные события, переданные объектом MDS. Менеджер должен поддерживать все методы, определенные в таблице 4.

MDS-Data-Request (Запрос данных MDS).

Данный метод позволяет системе менеджера для активации или дезактивации передачу измерительных данных от агента (см. 8.9.3.3.3 для подробного описания).

Set-Time (Установка времени).

Данный метод позволяет системе менеджера устанавливать часы реального времени (RTC) и абсолютного времени. Агент указывает, активна ли команда Set-Time при помощи mds-time-capab-set-clock bit в атрибуте Mds-Time-Info (см. таблицу 2).

Таблица 4 — События объектов MDS

Событие	Режим	Тип события	Параметр информации о событии	Информация об ответе на событие
MDS-Configuration-Event	Подтвержденный или неподтвержденный	MDC_NOTI_CONFIG	ConfigReport	ConfigReportRsp
MDS-Dynamic-Data-Update-Var	Подтвержденный или неподтвержденный	MDC_NOTI_SCAN_REPORT_VAR	ScanReportInfoVar	—
MDS-Dynamic-Data-Update-Fixed	Подтвержденный или неподтвержденный	MDC_NOTI_SCAN_REPORT_FIXED	ScanReportInfoFixed	—
MDS-Dynamic-Data-Update-MP-Var	Подтвержденный или неподтвержденный	MDC_NOTI_SCAN_REPORT_MP_VAR	ScanReportInfoMPVar	—
MDS-Dynamic-Data-Update-MP-Fixed	Подтвержденный или неподтвержденный	MDC_NOTI_SCAN_REPORT_MP_FIXED	ScanReportInfoMPFixed	—

Событие Конфигурации MDS (MDS-Configuration-Event).

Данное событие пересылается агентом во время настройки состояния запуска, если менеджер еще не знает конфигураций агента из предыдущих ассоциаций. Событие предоставляет статическую информацию о поддерживаемых измерительных возможностях агента.

Событие MDS-Dynamic-Data-Update-Var.

Данное событие предоставляет динамические данные (обычно измерения) от агента для нескольких или всех объектов, поддерживаемых агентом. Данные о сообщаемых объектах предоставляются, используя переменный формат списка атрибутов параметров настройки (см. 7.4.5 для более подробной информации или для форматов отчетов о событии). Событие запускается Запросом данных MDS-Data-Request из системы менеджера, или запрос пересылается как незапрашиваемое сообщение агентом. Для агентов, которые поддерживают инициированную менеджером передачу данных, см. 8.9.3.3.3 для получения дополнительной информации по контролю активации и/или периода передачи данных. Для информации по ограниченному контролю, который менеджер может подтверждать в отношении агентов, которые не поддерживают инициируемую менеджером передачу измерительных данных, см. 8.9.3.3.2.

Событие MDS-Dynamic-Data-Update-Fixe

Данное событие предоставляет динамические данные (обычно измерения от агента для нескольких или всех метрических объектов или объектов системы MDS, поддерживаемых агентом). Данные сообщаются в фиксированном формате, определенным атрибутом Attribute-Value-Map для сообщаемых метрических объектов или объектов системы MDS (см. 7.4.5 для подробной информации о форматах отчетов о событиях). Событие запускается Запросом данных системы MDS (MDS-Data-Request) от системы менеджера (т.е. инициируемая менеджером передача измерительных данных), или пересылается в качестве незапрашиваемого сообщения агентом (т. е. инициируемая агентом передача измерительных данных). Для агентов, которые поддерживают инициированную менеджером передачу данных, см. 8.9.3.3.3 для получения дополнительной информации по контролю активации и/или периода передачи данных. Для информации по ограниченному контролю, который менеджер может подтверждать в отношении агентов, которые не поддерживают инициируемую менеджером передачу измерительных данных, см. 8.9.3.3.2.

Событие MDS-Dynamic-Data-Update-MP-Var.

Данное событие сходится с MDS-Dynamic-Data-Update-Var, позволяя кроме того включение данных от нескольких лиц.

Событие MDS-Dynamic-Data-Update-MP-Fixed.

Данное событие сходится с MDS-Dynamic-Data-Update-Fixed, позволяя кроме того включение данных от нескольких лиц.

6.3.2.6 Другие сервисы системы MDS

6.3.2.6.1 Сервис GET (ПОЛУЧЕНИЕ)

Любой агент, поддерживающий линии двусторонней связи, должен поддерживать сервис GET (ПОЛУЧЕНИЕ) для извлечения значений всех применимых атрибутов объекта MDS. Сервис GET может быть запущен, как только агент получает Ответ на Ассоциацию и приобретает состояние Ассоциации, включая подсостояния Работа и Настройка.

Менеджер может запросить атрибуты объекта системы MDS агента, в случае которого менеджер должен переслать команду «Запрос удаленной работы|Получить» («Remote Operation Invoke | Get») (см. roiv-cmip-get в А.10.2) с зарезервированным значением дескриптора 0. Агент должен ответить, сообщая атрибуты своего объекта системы MDS менеджеру, используя команду ответа «Ответ Удаленной Работы|Получить» (см. rors-cmip-get в А.10.2). В ответ на команду Получить Объект MDS, возвращаются только примененные агентом атрибуты.

6.3.2.6.2 Сервис SET

На данный момент устанавливаемые атрибуты не представлены.

6.3.3 Метрический класс

6.3.3.1 Общие положения

Метрический класс является базовым классом для всех объектов, представляющим данные об измерениях, статусе и контекстные данные. Метрический класс не является инстанциированным; следовательно, он никогда не является частью конфигурации агента. Как базовый класс, он определяет все атрибуты, методы, события и сервисы, которые являются общими для всех объектов, представляющих измерения.

6.3.3.2 Идентификация метрического класса

Номенклатурный код для идентификации метрического класса: MDC_MOC_VMO_METRIC. Данный номенклатурный код не используется при применении агента или менеджера, так как метрический класс является лишь базовым классом для остальных классов.

6.3.3.3 Атрибуты метрического класса

Таблица 5 определяет набор метрических атрибутов, поддерживаемых для связи персональных медицинских приборов, и наследуемых всеми классами метрического происхождения.

Таблица 5 — Метрические атрибуты

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластор
Handle	MDC_ATTR_ID_ HANDLE	Handle	Атрибут Handle (дескриптор) представляет собой ссылочный ID для данного объекта. У каждого объекта должен быть уникальный ID, присвоенный агентом. Атрибут handle идентифицирует объект в отчетах о событии, пересылаемых менеджеру. Данный атрибут должен оставаться неизменным после одобрения конфигурации	M
Type	MDC_ATTR_ID_ TYPE	Type	Данный атрибут определяет особый статический этого объекта в соответствии с номенклатурой (например, частота импульсов для экземпляра числового объекта). Атрибут Type содержит номенклатурный раздел и ID кода термина для контекстно-свободной, расширяемой идентификации. Данный атрибут должен оставаться неизменным после одобрения конфигурации	M

Продолжение таблицы 5

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластер
Supplemental-Types	MDC_ATTR_ SUPPLEMENTAL_ TYPES	Supplemental-TypesList	Данный атрибут может использоваться для передачи дополнительной информации об объекте за рамками атрибутов Type и Metric-Id. Дополнительная информация покрывает такие условия, как расположение сенсора или уровень, при котором объект реагирует на изменения. Специализации прибора определяют предполагаемое применение данного атрибута. Например, ИИЭР Std 11073-10471 [5] определяет номенклатуру расположения, указывая расположение сенсора на дому и ИСО/ИИЭРР 11073-10404 [B9] определяет три дополнительных типа для быстрого ответа, медленного ответа и выборочной проверки частоты импульса или насыщенности крови кислородом. Данный атрибут должен оставаться неизменным после одобрения конфигурации	O
Metric-Spec-Small	MDC_ATTR_ METRIC_SPEC_ SMALL	Metric-Spec-Small	Данный атрибут описывает характеристики измерений	M
Metric-Structure-Small	MDC_ATTR_ METRIC_ STRUCT_SMALL	MetricStructureSmall	Данный атрибут описывает структуру измерения. В случае отсутствия менеджер должен принять MetricStructureSmall := {ms-struct-simple, 0}.	O
Measurement-Status	MDC_ATTR_ MSMT_STAT	MeasurementStatus	Данный атрибут указывает точность конкретного значения или набор считываний	O
Metric-Id	MDC_ATTR_ ID_PHYSIO	OID-Type	Данный атрибут может использоваться для удержания идентификации, которая более конкретна по сравнению с общим ID в атрибуте Type. Если оценивается атрибут Metric-Id-Partition, он определяет номенклатурный раздел для Данного атрибута. В противном случае, Тип OIDType совпадает с номенклатурным разделом, в соответствии с атрибутом Type в поле раздела. Данный атрибут требуется, только если меняется во время работы и атрибут Type не содержит полной идентификации. Например, если атрибут Type содержит общий температурный класс (MDC_TEMP), данный атрибут может сообщить отдельную, но изменяющуюся идентификацию MDC_TEMP_ORAL или MDC_TEMP_RECT. Должен присутствовать только один атрибут Metric-Id и Metric-Id-List	O

Продолжение таблицы 5

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кла-тор
Metric-Id-List	MDC_ATTR_ ID_PHYSIO_ LIST	MetricIdList	Данный атрибут должен использоваться, когда используется сложное наблюдаемое значение и включается напрямую Metric-Id (например, Compound-Simple-Nu-Observed-Value или Compound-Basic-Nu-Observed-Value) таким образом, чтобы элементы в списке наблюдаемых значений могли идентифицироваться индивидуально. Порядок списка Metric-Id-List должен соответствовать порядку элементов в сложном наблюдаемом значении. Должен присутствовать только один атрибут of Metric-Id и Metric-Id-List	C
Metric-Id-Partition	MDC_ATTR_ METRIC_ID_ PART	NomPartition	Данный атрибут может использоваться для определения раздела, из которого были взяты номенклатурные термины Metric-Id или Metric-Id-List. Если атрибут отсутствует, раздел совпадает с номенклатурным разделом, определенным в поле раздела атрибута Type	O
Unit-Code	MDC_ATTR_ UNIT_CODE	OID-Type	Данный атрибут определяет номенклатурный код для единиц измерения из раздела nom-partdim (например, MDC_DIM_KILO_G)	O
Attribute-Value-Map	MDC_ATTR_ ATTRIBUTE_ VAL_MAP	AttrValMap	Данный атрибут определяет атрибуты, о которых сообщается в сообщении обновления данных фиксированного формата. Использование этого атрибута обязательно, если агент использует сообщения фиксированного формата с значениями для сообщения данных динамического измерения для объектов	C
Source-Handle-Reference	MDC_ATTR_ SOURCE_ HANDLE_REF	HANDLE	Данный атрибут устанавливает отношение данного экземпляра объекта к первичному объекту. Данный атрибут используется, когда требуется смоделировать явное соотношение между экземплярами объекта для определения зависимостей. Использование данного атрибута определяется специализацией прибора	O
Label-String	MDC_ATTR_ ID_LABEL_ STRING	OCTET STRING	Данный атрибут определяет текстовое представление атрибута типа в печатном ASCII. Значение Данного атрибута полностью зависит от выбора производителя агента. Это может быть потенциально полезным для менеджера в качестве строки дисплея или в качестве помощи по выбору поведения, когда он не может понять MDC_ATTR_ID_TYPE отправленного агентом	O
Unit-Label-String	MDC_ATTR_ UNIT_LABEL_ STRING	OCTET STRING	Данный атрибут определяет текстовое представление размера кода блока Unit-Code в печатном ASCII. Значение Данного атрибута полностью зависит от выбора производителя агента. Это может быть потенциально полезным для менеджера в качестве строки дисплея или в качестве помощи по выбору поведения, когда он не может понять MDC_ATTR_UNIT_CODE, отправленного агентом	O

Окончание таблицы 5

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластер
Absolute-Time-Stamp	MDC_ATTR_TIME_STAMP_ABS	AbsoluteTime	Данный атрибут определяет дату и время наблюдения с точностью в 1/100 секунды, если это применимо. За более подробной информацией по Данному атрибуту обратитесь к п. 8.12. Если агент сохраняет данные (в объекте РМ-блока или как «временно хранимые измерения»), он должен ассоциировать отметку времени (Absolute-Time-Stamp, Relative-Time-Stamp или HiRes-Time-Stamp) с данными	С
Relative-Time-Stamp	MDC_ATTR_TIME_STAMP_REL	RelativeTime	Данный атрибут определяет время наблюдения (отметку времени в формате относительного времени/количества тика часов в соответствии с типом данных RelativeTime). Если агент сохраняет данные, он должен ассоциировать отметку времени (Absolute-Time-Stamp, Relative-Time-Stamp или HiRes-Time-Stamp) с данными	С
HiRes-Time-Stamp	MDC_ATTR_TIME_STAMP_REL_HI_RES	HighResRelative-Time	Данный атрибут определяет время наблюдения (отметку времени в формате высокоточного относительного времени/с количеством тактов часов в соответствии с типом данных HighResRelativeTime). Если агент хранит данные, он должен ассоциировать отметку времени (Absolute-Time-Stamp, Relative-Time-Stamp или HiRes-Time-Stamp) с данными	С
Measure-Active-Period	MDC_ATTR_TIME_PD_MSMT_ACTIVE	FLOAT-Type	Данный атрибут определяет длительность периода наблюдения в секундах	О

6.3.3.4 Методы метрического объекта

Нет

6.3.3.5 События метрического объекта

Объекты, которые происходят из метрического класса, не сообщают напрямую о своих наблюдениях; чаще всего наблюдения передаются посредством другого объекта, такого как объект системы MDS, объект сканера или объект РМ-блока.

6.3.3.6 Другие метрические сервисы

Нет

6.3.4 Числовой класс

6.3.4.1 Общие положения

Экземпляр числового класса представляет числовое измерение. Значения числового объекта пересылаются от агента менеджеру, используя сервис ОТЧЕТ О СОБЫТИЯХ (EVENT REPORT) (см. 7.3). Данный класс происходит от метрического базового класса.

6.3.4.2 Идентификация числового класса

Номенклатурный код для идентификации числового класса: MDC_MOC_VMO_METRIC_NU.

6.3.4.3 Атрибуты числового класса

Таблица 6 определяет набор числовых атрибутов, поддерживаемых для обмена данными персонального медицинского прибора.

Таблица 6 — Числовые атрибуты

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластер
Simple-Nu-Observed-Value	MDC_ATTR_NU_VAL_OBS_SIMP	SimpleNuObsValue	Данный атрибут определяет числовое наблюдаемое значение объекта без какой-либо вложенной информации о статусе, наблюдаемой в Nu-Observed-Value. Должно присутствовать одно из значений Simple-Nu-Observed-Value, Basic-Nu-Observed-Value, Nu-Observed-Value, Compond-Nu-Observed-Value, Compound-Simple-Nu-Observed-Value или Compound-Basic-Nu-Observed-Value	C
Compound-Simple-Nu-Observed-Value	MDC_ATTR_NU_CMPD_VAL_OBS_SIMP	SimpleNuObsValueCmp	Данный атрибут представляет матрицу Simple-Nu-Observed-Values. Должно присутствовать одно из значений Simple-Nu-Observed-Value, Basic-Nu-Observed-Value, Nu-Observed-Value, Compond-Nu-Observed-Value, Compound-Simple-Nu-Observed-Value или Compound-Basic-Nu-Observed-Value	C
Basic-Nu-Observed-Value	MDC_ATTR_NU_VAL_OBS_BASIC	BasicNuObsValue	Данный атрибут определяет числовое наблюдаемое значение объекта без какой-либо вложенной информации о статусе, но с небольшим числовым представлением, по сравнению с значением Simple-Nu-Observed-Value. Должно присутствовать одно из значений Simple-Nu-Observed-Value, Basic-Nu-Observed-Value, Nu-Observed-Value, Compound-Nu-Observed-Value,Compound-Simple-Nu-Observed-Value или Compound-Basic-Nu-Observed-Value	C
Compound-Basic-Nu-Observed-Value	MDC_ATTR_NU_CMPD_VAL_OBS_BASIC	BasicNuObsValueCmp	Данный атрибут представляет собой матрицу Basic-Nu-Observed-Values. Должно присутствовать одно из значений Simple-Nu-Observed-Value, Basic-Nu-Observed-Value, Nu-Observed-Value, Compond-Nu-Observed-Value, Compound-Simple-Nu-Observed-Value или Compound-Basic-Nu-Observed-Value	C
Nu-Observed-Value	MDC_ATTR_NU_VAL_OBS	NuObsValue	Данный атрибут определяет числовое наблюдаемое значение объекта и объединяет статус измерения и информацию о единице измерения. Он используется, когда статус/единица динамичные и всегда представляются вместе со значением. Должно присутствовать одно из значений Simple-Nu-Observed-Value, Basic-Nu-Observed-Value, Nu-Observed-Value, Compond-Nu-Observed-Value, Compound-Simple-Nu-Observed-Value или Compound-Basic-Nu-Observed-Value	C
Compound-Nu-Observed-Value	MDC_ATTR_NU_CMPD_VAL_OBS	NuObsValueCmp	Данный атрибут сочетает матрицу значения, статуса и единицы. Данный атрибут доступен для использования только в отчетах о событиях в переменном формате. Должно присутствовать одно из значений Simple-Nu-Observed-Value, Basic-Nu-Observed-Value,Nu-Observed-Value, Compond-Nu-Observed-Value,Compound-Simple-Nu-Observed-Value или Compound-Basic-Nu-Observed-Value	C

Окончание таблицы 6

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластор
Accuracy	MDC_ATTR_NU_ACCUR_MSMT	FLOAT-Type	Данный атрибут определяет максимальное отклонение текущего значения от сообщенного наблюдаемого значения (если он может быть определен). Данный атрибут должен оставаться неизменным после одобрения конфигурации	O

Атрибуты Compound-Simple-Nu-Obs-Value, Compound-Basic-Nu-Obs-Value и Compound-Nu-Observed-Value представляют список концептов для наблюдаемых значений. Данный концепт должен использоваться, когда между отдельными наблюдаемыми значениями дана сильная взаимосвязь, которая может быть зависима от номенклатуры и/или применения. Сложные наблюдаемые значения разделяют общий контекст статической атрибуции, за исключением идентификации элементов. Примером этого может служить применение артериального давления, где номенклатурный базовый термин выражает «Артериальное давление», а более специфические термины выражают «Артериальное давление Систолическое», «Артериальное давление Диастолическое», и «Артериальное давление Среднее». Соответствующий DIM должен содержать только один экземпляр числового объекта, который использовал бы один из форматов сложных числовых значений наблюдений для представления «систолической», «диастолической» и «средней» частей «Артериального давления».

6.3.4.4 Методы числового объекта

Нет

6.3.4.5 События числового объекта

Нет

6.3.4.6 Другие числовые сервисы

Нет

6.3.5 Класс RT-SA

6.3.5.1 Общие положения

Экземпляр класса RT-SA представляет измерение формы сигнала. Значения объекта RT-SA пересылаются от агента менеджеру с помощью сервиса ОТЧЕТ О СОБЫТИЯХ (EVENT REPORT) (см. 7.3). Данный класс происходит от метрического базового класса.

6.3.5.2 Идентификация класса RT-SA

Номенклатурный код для идентификации класса RT-SA: DC_MOC_VMO_METRIC_SA_RT.

6.3.5.3 Атрибуты класса RT-SA

Таблица 7 определяет набор атрибутов RT-SA, поддерживаемых для связи персонального медицинского устройства.

Показаний, снятых в режиме реального времени

Таблица 7 — Атрибуты RT-SA

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластор
Sample-Period	MDC_ATTR_TIME_PD_SAMP	RelativeTime	Данный атрибут определяет интервал времени между последовательными показаниями, осуществляющимися каждые 1/8 мс. Следовательно, 8000 = 1 с. Данный атрибут должен оставаться неизменным после одобрения конфигурации	M
Simple-Sa-Observed-Value	MDC_ATTR_SIMP_SA_OBS_VAL	OCTET STRING	Данный байтовый массив содержит показания, которые сообщаются агентом в формате, описанном в спецификации Sa-Specification и спецификации масштаба и диапазона (Scale-and-Range-Specification). Длина должна быть с дополняющими байтами в конце. Sa-Specification определяет текущее количество используемых байтов	M

Окончание таблицы 7

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кла-тор
Scale-and-Range-Specification	MDC_ATTR_SCALE_SPECN_I8 MDC_ATTR_SCALE_SPECN_I16 MDC_ATTR_SCALE_SPECN_I32	ScaleRangeSpec8 ScaleRangeSpec16 ScaleRangeSpec32	Данный атрибут определяет присвоение определенных значений показаниями, а также диапазон измерений. Тип зависит от разрешающей способности показания (поле размера показания в рамках поля типа показания в спецификации Sa-Specification). Должна быть включена одна из трех спецификаций. Данный атрибут должен оставаться неизменным после одобрения конфигурации	M
Sa-Specification	MDC_ATTR_SA_SPECN	SaSpec	Данный атрибут описывает массив показаний и типы показаний. Данный атрибут должен оставаться неизменным после одобрения конфигурации	M

Характеристика объекта RT-SA может быть получена путем исследования атрибута Sa-Specification. Данный атрибут определяет число элементов в матрице и размер элементов, более подробная информация дана в А.3.4.

Об агентах, которые поддерживают инициированную менеджером передачу данных, см. в 8.9.3.3.3 дополнительную информацию по контролю активации и/или периода передачи данных. Информацию по ограниченному контролю, который менеджер может подтверждать в отношении агентов, которые не поддерживают инициируемую менеджером передачу измерительных данных, см. в 8.9.3.3.2.

Атрибут Scale-and-Range-Specification.

Атрибут Scale-and-Range-Specification определяет коэффициент алгоритма для преобразования измеренных значений в их абсолютные значения. Менеджер должен применить следующий алгоритм:

$$Y = M \times X + B,$$

где Y — преобразованное абсолютное значение;

M = (наибольшее абсолютное значение — наименьшее абсолютное значение) / (наибольшее измеренное значение — наименьшее измеренное значение);

B = наибольшее абсолютное значение — (M × наибольшее измеренное значение);

X — измеренное значение.

Пример работы данного алгоритма приведен в приложении В.

Необходимо отметить, что в данном случае термин абсолютное значение не употребляется как математическое абсолютное значение, при котором все значения положительны, а скорее как реальное измеренное значение.

6.3.5.4 Методы объекта RT-SA

Нет

6.3.5.5 События объекта RT-SA

Нет

6.3.5.6 Другие сервисы RT-SA

Нет

6.3.6 Класс перечисления

6.3.6.1 Общие положения

Экземпляр класса перечисления представляет статусную информацию и/или аннотационную информацию. Значения объектов перечисления кодируются в форме нормативных кодов (в соответствии с ИСО/ИИЭР 11073-10101 [12]) или в свободной форме. Значения объекта перечисления пересылаются от агента менеджеру с помощью сервиса ОТЧЕТ О СОБЫТИЯХ (EVENT REPORT) (см. 7.3). Данный класс происходит от метрического базового класса.

6.3.6.2 Идентификация класса перечисления

Номенклатурный код для идентификации класса перечисления: MDC_MOC_VMO_METRIC_ENUM.

6.3.6.3 Атрибуты класса перечисления

Таблица 8 определяет набор атрибутов перечисления поддерживаемых для связи персонального медицинского устройства.

Таблица 8 — Атрибуты перечисления

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластор
Enum-Observed-Value-Simple-OID	MDC_ATTR_ENUM_OBS_VAL_SIMP_OID	OID-Type	Данное значение сообщается как номенклатурный код. Если атрибуту Enum-Observed-Value-Partition присваивается значение, он определяет номенклатурный раздел для данного атрибута. В противном случае, OID-Type берется из того же номенклатурного раздела, определенного в поле раздела атрибута Type. Должно присутствовать одно из значений Enum-Observed-Value-Simple-OID, Enum-Observed-Value-Simple-Bit-Str, Enum-Observed-Value-Basic-Bit-Str,Enum-Observed-Value-Simple-Str или Enum-Observed-Value	C
Enum-Observed-Value-Simple-Bit-Str	MDC_ATTR_ENUM_OBS_VAL_SIMP_BIT_STR	BITS-32	Данное значение сообщается в виде 32-битовой строки. Должно присутствовать одно из значений Enum-Observed-Value-Simple-OID, Enum-Observed-Value-Simple-Bit-Str, Enum-Observed-Value-Basic-Bit-Str, Enum-Observed-Value-Simple-Str или Enum-Observed-Value	C
Enum-Observed-Value-Basic-Bit-Str	MDC_ATTR_ENUM_OBS_VAL_BASIC_BIT_STR	BITS-16	Данное значение сообщается в виде 16-битовой строки. Должно присутствовать одно из значений Enum-Observed-Value-Simple-OID,Enum-Observed-Value-Simple-Bit-Str, Enum-Observed-Value-Basic-Bit-Str, Enum-Observed-Value-Simple-Str или Enum-Observed-Value	C
Enum-Observed-Value-Simple-Str	MDC_ATTR_ENUM_OBS_VAL_SIMP_STR	EnumPrintableString	Данное значение сообщается в виде ASCII печатной строки. Должен присутствовать один из Enum-Observed-Value-Simple-OID, Enum-Observed-Value-Simple-Bit-Str, Enum-Observed-Value-Basic-Bit-Str, Enum-Observed-Value-Simple-Str или Enum-Observed-Value	C
Enum-Observed-Value	MDC_ATTR_VAL_ENUM_OBS	EnumObsValue	Данный атрибут определяет структурное наблюдаемое значение, которое позволяет дополнительную гибкость типа данных сообщаемого значения. Должно присутствовать одно из значений Enum-Observed-Value-Simple-OID, Enum-Observed-Value-Simple-Bit-Str, Enum-Observed-Value-Basic-Bit-Str, Enum-Observed-Value-Simple-Str или Enum-Observed-Value	C
Enum-Observed-Value-Partition	MDC_ATTR_ENUM_OBS_VAL_PART	NomPartition	Данный атрибут может использоваться для определения раздела, из которого был взят наблюдаемый номенклатурный термин OID значений Enum-Observed-Value-Simple-OID или Enum-Observed-Value. В противном случае, OID-Type берется из того же номенклатурного раздела, определенного в поле раздела атрибута Type	O

6.3.6.4 Методы объекта перечисления

Нет

6.3.6.5 События объекта перечисления

Нет

6.3.6.6 Другие сервисы перечисления

Нет.

6.3.7 Класс РМ-блока

6.3.7.1 Общие положения

Экземпляр класса РМ-блока предоставляет возможности долгосрочного хранения метрических данных. Данные хранятся в переменном числе объектов РМ-сегмента (см. 6.3.8). Хранимые данные объекта РМ-блока запрашиваются от агента менеджером, используя сервисы доступа к объектам (см. 7.3). Если концепт РМ-блока неизвестен, можно обратиться к приложению С для концептуального обзора перед тем, как ознакомиться со следующими подпунктами.

6.3.7.2 Идентификация класса РМ-блока

Номенклатурный код для идентификации класса РМ-блока: MDC_MOC_VMO_PMSTORE.

6.3.7.3 Атрибуты класса РМ-блока

Таблица 9 определяет набор атрибутов РМ-блока, поддерживаемых для связи персонального медицинского устройства:

Таблица 9 — Атрибуты РМ-блока

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластор
Handle	MDC_ATTR_ID_ HANDLE	HANDLE	Атрибут Handle (дескриптор) представляет собой ссылочный ID для данного объекта. У каждого объекта должен быть свой ID, предписанный агентом. Дескриптор идентифицирует объект в отчете о событиях, пересылаемом менеджеру и на адрес экземпляра объекта в сообщении, запускающем методы объекта. Данный атрибут должен оставаться неизменным после одобрения конфигурации	M
PM-Store-Capab	MDC_ATTR_ PM_STORE_ CAPAB	PmStoreCapab	Данный атрибут определяет базовые возможности экземпляра объекта РМ-блока. Данный атрибут должен оставаться неизменным после одобрения конфигурации	M
Store-Sample-Algorithm	MDC_ATTR_ METRIC_ STORE_ SAMPLE_ALG	StoSampleAlg	Данный атрибут описывает, как обрабатываются значения считываний, хранимых в РМ-сегменте. Структура StoSampleAlg описывает доступные алгоритмы выборки. Если не применяется специфического алгоритма выборки (другими словами, значения считывания являются необработанными данными), то данный атрибут должен иметь значение st-alg-no-downsampling. Данный атрибут должен оставаться неизменным после одобрения конфигурации	M
Store-Capacity-Count	MDC_ATTR_ METRIC_ STORE_ CAPAC_CNT	INT-U32	Данный атрибут это максимальное количество хранимых записей в РМ-сегменте (записи во всех сегментах РМ). Данный атрибут должен оставаться неизменным после одобрения конфигурации	O
Store-Usage-Count	MDC_ATTR_ METRIC_STORE_ USAGE_CNT	INT-U32	Данный атрибут дает текущее (актуальное) количество хранимых записей РМ-сегмента (записи во всех сегментах РМ)	O
Operational-State	MDC_ATTR_ OP_STAT	OperationalState	Данный атрибут указывает, вносятся ли в данный момент новые записи в этот РМ-сегмент. Если в данный момент РМ-сегмент добавляет новые данные, данный атрибут должен быть активирован. В противном случае, он должен дезактивироваться	M
PM-Store-Label	MDC_ATTR_ PM_STORE_ LABEL_STRING	OCTET STRING	Данный атрибут является зависящей от применения отметкой в печатном ASCII для сегмента, чтобы указать его предполагаемое назначение, а также может использоваться для отображения целей. Данный атрибут должен оставаться неизменным после одобрения конфигурации	O

Окончание таблицы 9

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кла-тор
Sample-Period	MDC_ATTR_ TIME_PD_SAMP	RelativeTime	Данный атрибут определяет частоту, с которой в PM-сегменты добавляются новые записи. Данный атрибут должен присутствовать либо в PM-блоке (в случае чего он относится ко всем периодически хранимым PM-сегментам в PM-блоке), либо в каждом PM-сегменте, если значения отбираются периодически, таким образом разница во времени двух записей в фиксированном сегменте данных постоянна (т.е. установлен бит записей pmsc-periseg-entries в атрибут Pm-Store-Capab). Данный атрибут должен оставаться неизменным после одобрения конфигурации	C
Number-Of-Segments	MDC_ATTR_ NUM_ SEG	INT-U16	Данный атрибут это число актуально инстанцируемых PM-сегментов, содержащихся в PM-блоке. Необходимо отметить, что Номер экземпляра Instance-Number атрибута PM-сегмента НЕ относится к этому числу (т.е. он не должен находиться в диапазоне от 0 до Числа сегментов), но должен извлекаться методом Get-Segment-Info	M
Clear-Timeout	MDC_ATTR_ CLEAR_ TIMEOUT	RelativeTime	Данный атрибут тайм-аута определяет минимальное время, в течение которого менеджер должен ожидать завершения команды очистки PM-блока. Если после того, как менеджер пересылает запуск команды Подтвержденное Действие (Очистить сегменты), истекает тайм-аут до того, как менеджер получает соответствующее сообщение ответа команды Подтвержденное действие, менеджер должен осуществить переход в Неассоциированное состояние в соответствии с 8.9.5.6	M

Атрибуты Handle и PM-Store-Capab являются частью конфигурации агента; следовательно, менеджер знает соответствующие значения атрибута после процедуры конфигурации.

6.3.7.4 Методы объекта PM-блока

Таблица 10 определяет методы (действия) объекта PM-блока. Данные методы могут быть запущены, используя сервис ACTION.

Таблица 10 — Методы объекта PM-блока

Метод/действие	Режим	Тип действия	action-info-args	Resulting action-info-args
Clear-Segments	подтвержденный	MDC_ACT_SEG_CLR	SegmSelection	(пусто)
Get-Segment-Info C	подтвержденный	MDC_ACT_SEG_ GET_INFO	SegmSelection	SegmentInfoList
Trig-Segment-Data-Xfer	подтвержденный	MDC_ACT_SEG_ TRIG_XFER	TrigSegmDataXfer-Req	TrigSegmDataXferRsp

Если агент поддерживает класс PM-блока, поддержка методов Get-Segment-Info и Trig-Segment-Data-Xfer обязательна. Поддержка метода Clear-Segments является условной и указывается в атрибуте PM-Store-Capab.

Метод очистка сегментов (Clear-Segments)

Данный метод позволяет менеджеру удалять данные, хранимые в одном или нескольких сегментах PM. Все записи в выбранных сегментах PM удаляются. Если агент поддерживает переменное число PM-сегментов, агент может удалить пустые PM-сегменты. Кроме того, агент может очистить PM-сегменты без команды от менеджера (например, пользователь агента может выбрать удалить хранимые агентом

данные); однако, если эта операция производится в ассоциированном состоянии, номер экземпляра (Instance-Number) должен оставаться пустым в ходе ассоциации. Номер экземпляра (Instance-Number) всех остальных РМ-сегментов должен оставаться нетронутым при чистке сегмента. Если данный метод запускается на РМ-сегменте, у которого атрибут Operational-State активирован, агент должен ответить ошибкой not-allowed-by-object error (roer) с кодом возврата MDC_RET_CODE_OBJ_BUSY.

Необходимо отметить, что поведение метода Clear-Segments ориентировано на конкретное приложение. Метод может очистить все записи от предусмотренного РМ-сегмента, оставляя его пустым или может полностью очистить указанный РМ-сегмент. Данное поведение определяется в атрибуте PM-Store-Capab. Для определенных приложений рекомендации определяются в специализациях соответствующих приборов, применяющих РМ-блок.

Метод Get-Segment-Info

Данный метод позволяет менеджеру извлечь атрибуты РМ-сегмента одного или нескольких РМ-сегментов, за исключением данных атрибута фиксированного сегмента, который содержит актуальные данные и извлекается при помощи метода Trig-Segment-Data-Xfer. В частности, метод Get-Segment-Info позволяет менеджеру извлечь атрибуты Instance-Numbe экземпляров объекта РМ-сегмента и содержание их данных.

Метод Trig-Segment-Data-Xfer

Данный метод позволяет менеджеру начать передачу данных атрибута фиксированного сегмента определенного РМ-сегмента. Агент указывает в ответ, принимает ли он или отклоняет этот запрос. Если агент принимает запрос, агент посылает сообщения Segment-Data-Event в соответствии с описанием в 6.3.7.5. Если данный метод запускается на РМ-сегменте, у которого активирован атрибут Operational-State, агент должен ответить ошибкой not-allowed-by-object error (roer) с кодом возврата MDC_RET_CODE_OBJ_BUSY.

6.3.7.5 События объекта РМ-блока

Таблица 11 определяет потенциальные события, пересылаемые объектом РМ-блока:

Таблица 11 — Событие объекта РМ-блока

Событие	Режим	Тип события	Параметр информации о событии	Информация об ответе
Segment-Data-Event	подтвержденный	MDC_NOTI_ SEGMENT_DATA	SegmentDataEvent	SegmentDataResult

Событие Segment-Data-Event

Данное событие посылает хранимые в фиксированном сегменте РМ-сегмента данные от агента менеджеру. Данное событие запускается менеджером методом Trig-Segment-Data-Xfer. Как только передача данных запускается, агент посылает сообщения Segment-Data-Event до тех пор, пока данные фиксированного сегмента полностью не будут переданы или если передача не будет отменена менеджером или агентом. Для более полной информации о содержании передачи РМ-сегмента обратитесь к п. 8.9.3.4.2.

Рекомендуется размещать как можно больше записей сегмента, содержащихся в Segment-Data-Event для сокращения количества сообщений, требуемых для передачи сегмента.

Поддержка события агентом обязательна, если агент поддерживает объекты РМ-блока.

6.3.7.6 Другие сервисы РМ-блока

6.3.7.6.1 Сервис GET

Поддержка сервиса GET должна обеспечиваться любым агентом, поддерживающим один и более объектов РМ-блока только во время состояния работы. Менеджер использует сервис GET для извлечения значений всех атрибутов объекта РМ-блока.

Менеджер может запросить атрибуты объекта РМ-блока агента, в случае чего менеджер должен послать команду «Remote Operation Invoke | Get» (см. roiv-cmip-get в А.10.2) со значением дескриптора объекта РМ-блока, определенного в конфигурации агента. Агент должен ответить, сообщая свои атрибуты объекта РМ-блока менеджеру, используя ответ «Remote Operation Response | Get» (см. rors-cmip-get в А.10.2).

6.3.8 Класс РМ-сегмента

6.3.8.1 Общие положения

Экземпляр класса РМ-сегмента представляет собой постоянно хранимые примеры измерительных данных. Объект РМ-сегмента не является частью конфигурации статического агента, т. к. количество экземпляров инстанциированного РМ-сегмента может динамично меняться. Менеджер осуществляет опосредованный доступ в объекты РМ-сегмента посредством методов и событий объекта РМ-блока.

6.3.8.2 Идентификация класса РМ-сегмента

Номенклатурный код для идентификации класса РМ-сегмента: MDC_MOC_PM_SEGMENT.

6.3.8.3 Атрибуты класса РМ-сегмента

Таблица 12 определяет набор атрибутов РМ-сегмента, поддерживаемых для связи персонального медицинского устройства.

Таблица 12 — Атрибуты РМ-сегмента

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кла-тор
Instance-Number	MDC_ATTR_ ID_INSTNO	InstNumber	Номер экземпляра (Instance-Number) это ID определенного экземпляра объекта РМ-сегмента. Он используется менеджером для обращения к сегменту РМ	M
PM-Segment-Entry-Map	MDC_ATTR_ PM_SEG_ MAP	PmSegmentEntryMap	Данный атрибут определяет формат и содержание одной сохраненной записи. Запись обладает условным заголовком, содержащим информацию, применимую ко всем элементам в записи. Запись содержит один и более элементов, определяемых по классу, метрическому ID, дескриптору и карте значения атрибута, определяющей атрибуты объекта для каждого элемента в сегменте РМ	M
PM-Seg-Person-Id	MDC_ATTR_ PM_SEG_ PERSON_ID	PersonId	Настоящий стандарт поддерживает приборы, которые обладают поддержкой простых данных от нескольких лиц. ID лица используется для различия этих лиц. Если РМ-блок способен сохранять данные нескольких лиц, он должен установить бит mscmultiperson в атрибуте PMStore-Capab. При установке этого бита все экземпляры РМ-сегмента, содержащихся в РМ-блоке, должны поддерживать атрибут PM-Seg-Person-Id. В противном случае данный атрибут не определяется	C
Operational-State	MDC_ATTR_ OP_STAT	OperationalState	Данный атрибут указывает, вносятся ли в данный момент новые записи в этот РМ-сегмент. Если в данный момент РМ-сегмент добавляет новые данные, данный атрибут должен быть активирован. В противном случае, он должен дезактивироваться	M
Sample-Period	MDC_ATTR_ TIME_PD_ SAMP	RelativeTime	Данный атрибут определяет частоту, с которой добавляются записи РМ-сегмента. Данный атрибут должен присутствовать либо в РМ-блоке (в случае чего он применяется ко всем периодически хранимым РМ-сегментам в РМ-блоке), либо в каждом РМ-сегменте. Значения дискретные, в атрибуте PM Store-Capab должен быть установлен бит записей pmsc-periseg-entries	C

Продолжение таблицы 12

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кла-тор
Segment-Label	MDC_ATTR_ PM_SEG_ LABEL_ STRING	OCTET STRING	Данный атрибут является зависящей от применения отметкой в печатном ASCII для сегмента, чтобы указать его предполагаемое назначение, может использоваться для отображения целей	O
Segment-Start-Abs-Time	MDC_ATTR_ TIME_START_ SEG	AbsoluteTime	Данный атрибут определяет начальное время сегмента	O
Segment-End-Abs-Time	MDC_ATTR_ TIME_END_ SEG	AbsoluteTime	Данный атрибут определяет конечное время сегмента	O
Date-and-Time-Adjustment	MDC_ATTR_ TIME_ABS_ ADJUST	AbsoluteTimeAdjust	Данный атрибут сообщает о любых изменениях даты и времени, которые вносятся либо в связи с изменением часов лица или переходом на летнее время. Если агент когда-либо изменяет дату и время, данный атрибут сообщает о таких изменениях	C
Segment-Usage-Count	MDC_ATTR_ SEG_USAGE_ CNT	INT-U32	Данный атрибут дает текущее (актуальное) количество хранимых записей	O
Segment-Statistics	MDC_ATTR_ SEG_STATS	SegmentStatistics	Данный атрибут определяет матрицу для сообщения минимальной, средней и максимальной статистики для каждого элемента под маркировку	O
Fixed-Segment-Data	MDC_ATTR_ SEG_FIXED_ DATA	Не применимо. Эти данные хранятся внутри в устройстве и, таким образом, этот тип данных никогда не появляется непосредственно ни в каком определении протокола	Данный атрибут определяет данные сегмента, переданные в качестве матрицы записей в формате, указанном атрибутом PM-Segment-Entry-Map. Здесь он определен в качестве прозрачной структуры данных без определенного типа данных. Обратите внимание, что данный атрибут доступен опосредованно; он извлекается менеджером при помощи метода PM-блока Trig-Segm-Data-Xfer	M
Confirm-Timeout	MDC_ATTR_ CONFIRM_ TIMEOUT	RelativeTime	Данный атрибут информационного тайм-аута определяет минимальное время, в течение которого агент должен ожидать сообщения ответа от менеджера после выдачи сообщений запроса Подтвержденного Отчета о Событиях до тайм-аута и перевода в неассоциированное состояние. Это информационный атрибут в пользу менеджера. Если поставляется данный атрибут, он должен соответствовать текущему значению тайм-аута, который используется агентом для Подтвержденного Отчета о Событиях, выдаваемого объектом PM-блока. Данный атрибут является информационным для менеджера в том смысле, что менеджер не использует данный атрибут в текущей реализации протокола (т. е. менеджер не производит тайм-аут на Подтвержденный Отчет о Событиях, сгенерированный агентом). Однако менеджер может пожелать использовать данную информацию для назначения приоритетов обработки агента «короткого» тайм-аута над агентом «долгого» тайм-аута	O

Окончание таблицы 12

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластор
Transfer-Timeout	MDC_ATTR_ TRANSFER_ TIMEOUT	RelativeTime	Данный атрибут тайм-аута определяет минимальное время, в течение которого менеджер ждет полной передачи информации PM-сегмента. Если тайм-аут закончится до получения полного PM-сегмента, менеджер должен совершить передачу в неассоциированном состоянии в соответствии с описанием в п. 8.9.5.6	M

Атрибут Fixed-Segment-Data хранит матрицу записей единого формата. В случаях, когда измерение не может быть предоставлено за требуемый период времени, значение для числового измерения предоставленного типом данных (S) FLOAT-type должно использовать специальное значение NaN (не число), чтобы указать на то, что значение не доступно.

Атрибут Fixed-Segment-Data может удерживать достаточно большие объемы данных в зависимости от возможностей и применения агента. Агент может ограничить максимальный размер атрибута Fixed-Segment-Data, чтобы таким образом выровнять его с максимальным размером блока передачи транспортной системы. Чтобы поддерживать данный тип поведения, менеджер должен быть в состоянии поддерживать передачу атрибутов Fixed-Segment-Data сообщений многократного использования.

6.3.8.4 Методы объекта PM-segment

Нет

6.3.8.5 События объектов PM-segment

Нет

6.3.8.6 Другие сервисы PM-segment

Нет

6.3.9 Классы сканера

6.3.9.1 Общая информация

Объект сканера является наблюдателем и «сумматором» значений атрибутов объекта. Он наблюдает атрибуты метрических объектов (например, числовые объекты) и генерирует сводки в форме отчетов о событиях. См. на рисунке 5 иерархию классов сканера. Каждый класс описан в 6.3.9.2—6.3.9.5 соответственно.

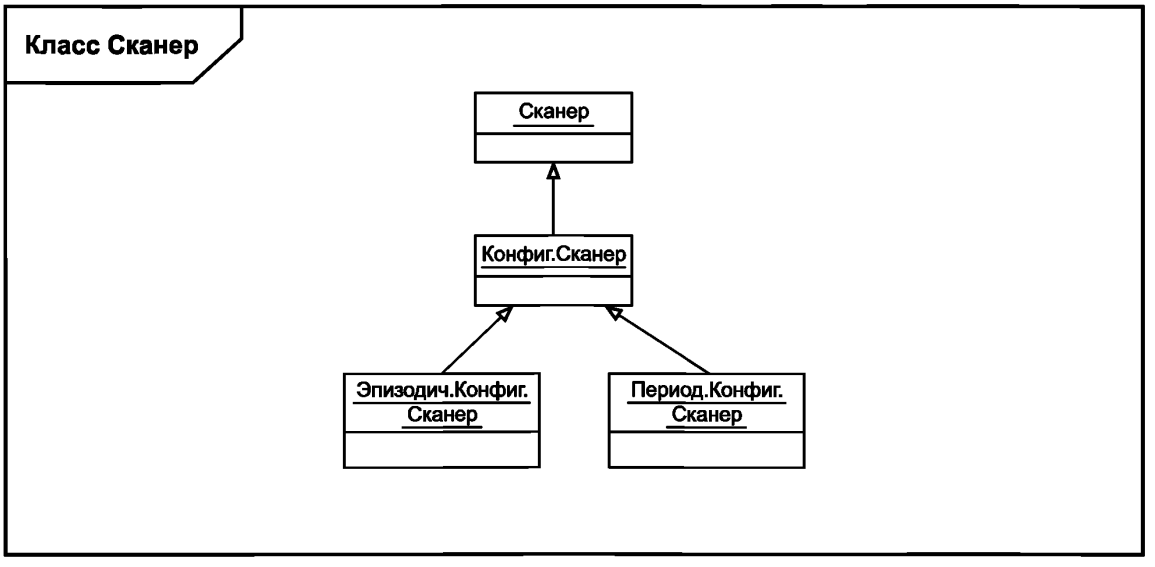


Рисунок 5 — Персональный медицинский прибор. DIM. Модель сканера

Должны использоваться различные классы сканера (периодические и эпизодические), а также различные экземпляры для распространения данных различных типов по одному или нескольким потокам данных, представленных экземпляром сканера. Приложение пульсовой оксиметрии может выбрать периодический конфигурируемый сканер для объектов RT-SA с интервалом отправки отчета каждые 50 мс, периодический конфигурируемый сканер с интервалом сообщения отчета в 1 сек для числовых объектов и объектов перечисления, а также конфигурируемый эпизодический сканер для импульсных метрических объектов (числовых объектов и объектов перечисления).

6.3.9.2 Класс сканера

6.3.9.2.1 Общая информация

Класс сканера является абстрактным классом, определяющим атрибуты, методы, события и сервисы, которые являются типичными для его подклассов. Будучи таковым, он не может инстанциироваться.

Концепт сканера обеспечивает три разных сообщения отчета о событиях: переменный формат, фиксированный формат и сгруппированный формат. См. в 7.4.5 информацию о сообщениях атрибутов наблюдаемых объектов. Форматы отчетов о событиях описаны ниже в 6.3.9.4.5, 6.3.9.5.5 и А.11.5, соответственно.

Более специализированные классы сканера происходят от базового класса сканера.

6.3.9.2.2 Идентификация класса сканера

Номенклатурный код для идентификации класса сканера: MDC_MOC_SCAN.

6.3.9.2.3 Атрибуты класса сканера

Таблица 13 определяет набор атрибутов сканера, поддерживаемых для связи персонального медицинского устройства.

Таблица 13 — Атрибуты сканера

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластор
Handle	MDC_ATTR_ID_ HANDLE	HANDLE	Сканеры идентифицируются дескрипторами (handles). Данный атрибут должен оставаться неизменным после одобрения конфигурации	M
Operational-State	MDC_ATTR_ OP_STAT	OperationalState	Данный атрибут определяет, активен ли сканер и может ли он быть установлен менеджером	M
Scan-Handle-List	MDC_ATTR_ SCAN_ HANDLE_LIST	HANDLEList	Данный атрибут определяет объекты с метрическим происхождением, которые могут сообщаться в отчетах Unbuf-Scan-Report-Var, Buf-Scan-Report-Var, Unbuf-Scan-Report-Fixed, Buf-Scan-Report-Fixed или в любом из четырех эквивалентов нескольким лицам. В случае эпизодических сканеров, специальный объект включается в отчет о событиях, как только вносятся изменения одного или нескольких значений атрибута. В случае периодических сканеров, сканируемые объекты и значения атрибутов вносятся в каждый из периодов. Менеджер не должен принимать порядок объектов, содержащихся в отчетах о событиях. Порядок совпадает со списком Scan-Handle-List. Данный атрибут должен предоставить информацию, используется ли сканером какой-либо из восьми стилей отчетов	C
Scan-Handle-Attr-Val-Map	MDC_ATTR_ SCAN_ HANDLE_ATTR_ VAL_MAP	HandleAttrValMap	Данный атрибут определяет метрические производные объекты, атрибуты и порядок, значения объектов и атрибутов которого сообщаются в Unbuf-Scan-Report-Grouped, Buf-Scan-Report-Grouped, Unbuf-Scan-Report-MP-Grouped или Buf-Scan-Report-MP-Grouped. Должны присутствовать все значения для обеспечения соответствующей структуры сообщения. Данный атрибут должен присутствовать, если используется один из этих четырех стилей	C

6.3.9.2.4 Методы объектов сканера

Нет

6.3.9.2.5 События объекта сканера

См. описания события производного класса в 6.3.9.4.5 и 6.3.9.5.5.

6.3.9.2.6 Другие сервисы сканера

Сервис SET

Агенты, которые обладают производными объектами сканера должны поддерживать набор сервисов атрибута Operational-State объектов сканера.

6.3.9.3 Класс CfgScanner

6.3.9.3.1 Общая информация

Класс CfgScanner является абстрактным классом, определяющим атрибуты, методы, события и сервисы, которые являются типичными для его подклассов. В частности, он определяет коммуникационное поведение объекта настраиваемого сканера. Будучи таковым, он не может инстанцироваться.

6.3.9.3.2 Идентификация класса конфигурируемого сканера

Номенклатурный код для идентификации класса конфигурируемого сканера: MDC_MOC_SCAN_CFG.

6.3.9.3.3 Атрибуты класса конфигурируемого сканера

Таблица 14 определяет набор атрибутов сканера, поддерживаемых для связи персонального медицинского устройства.

Таблица 14 — Атрибуты конфигурируемого сканера

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластер
Confirm-Mode	MDC_ATTR_CONFIRM_MODE	ConfirmMode	Данный атрибут определяет, являются ли пересланные отчеты о событиях подтвержденными или неподтвержденными	M
Confirm-Timeout	MDC_ATTR_CONFIRM_TIMEOUT	RelativeTime	Данный информационный тайм-аут атрибут определяет минимальное время, которое агент должен ожидать сообщения ответа от менеджера после выпуска Подтвержденного Отчета о Событиях до тайм-аута и перевода в неассоциированное состояние. Он является информационным атрибутом в пользу менеджера. Если данный атрибут поставляется, он должен подходить значению текущего тайм-аута, который использует агент для Подтвержденного Отчета События, заданного от объекта сканера. Данный атрибут является информационным для менеджера в том смысле, что менеджер не использует данный атрибут в текущем применении протокола (т.е. менеджер не устанавливает тайм-аут на Подтвержденный Отчет о Событии, сгенерированный агентом). Однако менеджер может пожелать использовать данную информацию для назначения приоритетов обработки агента «короткого» тайм-аута над агентом «долгого» тайм-аута	C
Transmit-Window	MDC_ATTR_TX_WIND	INT-U16	Данный атрибут определяет информационные данные, предоставляемые агентом, которые могут помочь менеджеру оптимизировать свои конфигурации. Окно передачи (Transmit-Window) представляет число неизвестных подтвержденных отчетов о событиях, которым агент позволит быть в очереди. Для настоящей версии данного стандарта, значение атрибута должно составлять 1	O

Рисунок 6 демонстрирует выполнение вспомогательной очереди передачи, когда значение Transmit-Window более чем 1.

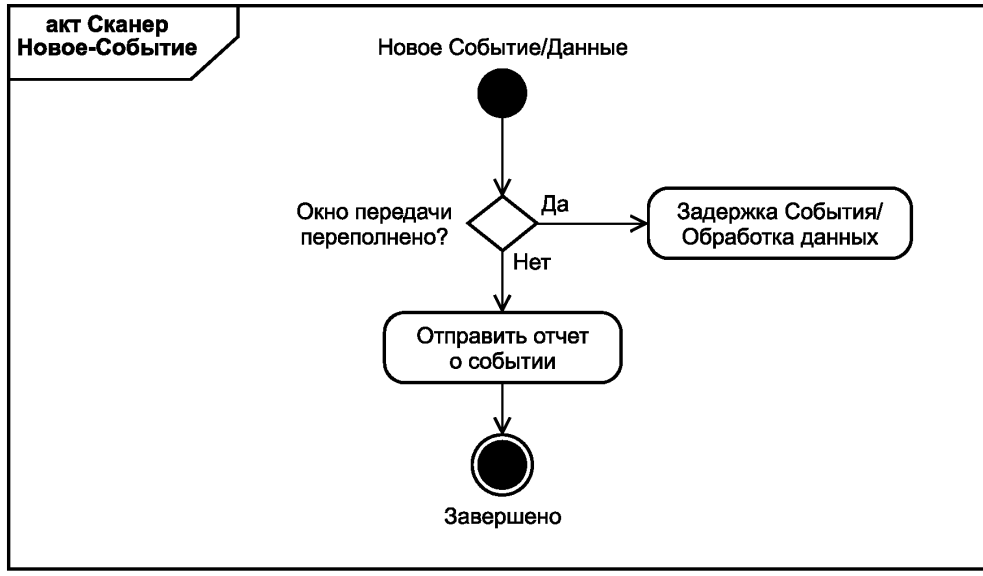


Рисунок 6 — Работа окна передачи (Transmit-Window) конфигурируемого сканера

6.3.9.3.4 Методы объекта конфигурируемого сканера

Нет

6.3.9.3.5 События объекта конфигурируемого сканера

См. описания событий производного класса в 6.3.9.4.5 и п. 6.3.9.5.5.

6.3.9.3.6 Другие события конфигурируемого сканера

Нет

6.3.9.4 Класс EpiCfgScanner

6.3.9.4.1 Общие положения

Класс EpiCfgScanner представляет собой класс, который может инстанцироваться. Объекты EpiCfgScanner используются для отправки отчетов, содержащих эпизодические данные, то есть данные, не имеющие фиксированного периода между каждым значением данных. Отчет пересылается, как только один из наблюдаемых атрибутов меняет значение; однако временной интервал между двумя последовательными отчетами о событиях не должен быть меньше значения атрибута Min-Reporting-Interval.

6.3.9.4.2 Идентификация класса эпизодического конфигурируемого сканера

Номенклатурный код для идентификации класса эпизодического конфигурируемого сканера: MDC_MOC_SCAN_CFG_EPI.

6.3.9.4.3 Атрибуты класса эпизодического конфигурируемого сканера

Таблица 15 определяет набор атрибутов класса эпизодического конфигурируемого сканера, поддерживаемых для связи персонального медицинского устройства.

Таблица 15 — Атрибуты эпизодического конфигурируемого сканера

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кла-тор
Min-Reporting-Interval	MDC_ATTR_SCAN_REP_PD_MIN	RelativeTime	Данный атрибут предоставляет расчет ожидаемого минимального времени между двумя последовательными отчетами о событиях	O

6.3.9.4.4 Методы объекта эпизодического конфигурируемого сканера

Нет

6.3.9.4.5 События объекта эпизодического конфигурируемого сканера

Таблица 16 определяет потенциальные события, пересылаемые объектом эпизодически конфигурируемого сканера. Отчеты о событиях классифицируются как безбуферные, т. к. агент пересылает событие, когда происходит эпизод, и он не должен буферизовать информацию в ожидании следующей периодической передачи. Если агент поддерживает эпизодический конфигурируемый сканер, он должен поддерживать как минимум одно из событий, указанных в таблице 16.

Таблица 16 — События объекта эпизодического конфигурируемого сканера

Событие	Режим	Тип события	Параметр информации о событиях	Информация об ответе
Unbuf-Scan-Report-Var	подтвержденный или неподтвержденный	MDC_NOTI_UNBUF_SCAN_REPORT_VAR	ScanReportInfoVar	-
Unbuf-Scan-Report-Fixed	подтвержденный или неподтвержденный	MDC_NOTI_UNBUF_SCAN_REPORT_FIXED	ScanReportInfoFixed	-
Unbuf-Scan-Report-Grouped	подтвержденный или неподтвержденный	MDC_NOTI_UNBUF_SCAN_REPORT_GROUPED	ScanReportInfoGrouped	-
Unbuf-Scan-Report-MP-Var	подтвержденный или неподтвержденный	MDC_NOTI_UNBUF_SCAN_REPORT_MP_VAR	ScanReportInfoMPVar	-
Unbuf-Scan-Report-MP-Var	подтвержденный или неподтвержденный	MDC_NOTI_UNBUF_SCAN_REPORT_MP_FIXED	ScanReportInfoMPFixed	-
Unbuf-Scan-Report-MP-Grouped	подтвержденный или неподтвержденный	MDC_NOTI_UNBUF_SCAN_REPORT_MP_GROUPED	ScanReportInfoMPGrouped	-

Примечания

1 В случае переменных и фиксированных форматов отчетов, если ни один из атрибутов объекта не меняет своего значения, в отчете сканирования не отображается каких-либо данных этого объекта.

2 В связи с тем, что эпизодический сканер не буферизует какие-либо изменения и не оснащен атрибутами периодического обновления спецификации (в этом нет необходимости, т. к. пересылаются обновления в случае внесения изменений), должно пересылаться уведомление об изменении атрибута, чтобы гарантировать отсутствие потери данных. Например, чтобы гарантировать, что ни одно из метрических значений не изменялось более одного раза между проведением сканирования атрибутов объекта, применение эпизодического сканера позволяет проверить изменения со скоростью, равной скорости самого короткого периода обновления метрических экземпляров в списке сканирования.

Событие Unbuf-Scan-Report-Var

Стиль данного события сообщает обобщенные данные о любом объекте и атрибуте, контролируемых сканером. Событие запускается, как только значения данных меняются, используется формат переменного сообщения (тип/длина/значение) для сообщения измененных данных.

Событие Unbuf-Scan-Report-Fixed

Стиль данного события используется, как только значения данных меняются, используется формат фиксированного сообщения каждого объекта для сообщения измененных данных.

Событие Unbuf-Scan-Report-Grouped

Стиль данного события используется, когда используется объект сканера для пересылки данных в их наиболее компактном формате. Атрибут Handle-Attr-Val-Map описывает включенные объекты и атрибуты, а также формат сообщения.

Событие Unbuf-Scan-Report-MP-Var

Стиль данного события полностью совпадает с Unbuf-Scan-Report-Var, но позволяет также включение данных от нескольких лиц.

Событие Unbuf-Scan-Report-MP-Fixed

Стиль данного события полностью совпадает с Unbuf-Scan-Report-Fixed, но позволяет также включение данных от нескольких лиц.

Событие Unbuf-Scan-Report-MP-Grouped

Стиль данного события полностью совпадает Unbuf-Scan-Report-Grouped, но позволяет также включение данных от нескольких лиц.

6.3.9.4.6 Другие сервисы эпизодически конфигурируемого сканера

Нет.

6.3.9.5 Класс PeriCfgScanner

6.3.9.5.1 Общие положения

Класс PeriCfgScanner представляет собой класс, из которого могут быть созданы экземпляры. Объекты PeriCfgScanner используются для пересылки отчетов, содержащих периодические данные, то есть данные, полученные в установленные периоды. Он буферизует любые изменения значений данных, которые должны пересылаться как часть периодического отчета. Отчеты о событиях должны пересылаться с временным интервалом, равным значению атрибута Reporting-Interval.

Количество наблюдений для каждого метрического объекта зависит от интервала обновления метрического объекта и интервала сообщения отчета сканера.

Пример — Периодически конфигурируемый сканер установлен на «сканирование» двух метрических объектов с интервалом сообщения отчетов в 1 сек. Два объекта периодически обновляют их соответствующие наблюдаемые значения с интервалом в 1 сек и $^1/_2$ сек, соответственно. Затем периодически конфигурируемый сканер выпускает отчеты о событиях каждую секунду, сообщая один скан наблюдения метрического объекта #1 и два скана наблюдения метрического объекта #2.

6.3.9.5.2 Идентификация объекта периодического конфигурируемого сканера

Номенклатурный код для идентификации класса периодического конфигурируемого сканера: MDC_MOC_SCAN_CFG_PERI.

6.3.9.5.3 Атрибуты объекта периодического конфигурируемого сканера

Таблица 17 определяет набор атрибутов объекта сканера, поддерживаемых для связи персонального медицинского устройства.

Таблица 17 — Атрибуты объекта периодического конфигурируемого сканера

Название атрибута	ID атрибута	Тип атрибута	Комментарий	Кластор
Reporting-Interval (интервал отправки отчета)	MDC_ATTR_SCAN_REP_PD	RelativeTime	Период сообщения отчета о событии	M

6.3.9.5.4 Методы объекта периодического конфигурируемого сканера

Нет

6.3.9.5.5 События объекта периодического конфигурируемого сканера

Таблица 18 определяет потенциальные события, пересылаемые объектом периодического конфигурируемого сканера. Если агент поддерживает периодический конфигурируемый сканер, он должен поддерживать как минимум события, перечисленные в таблице 18.

Таблица 18 — События объекта периодического конфигурируемого сканера

Событие	Режим	Тип события	Event-info parameter	Event-reply-info
Buf-Scan-Report-Var	Подтвержденный или неподтвержденный	MDC_NOTI_BUF_SCAN_REPORT_VAR	ScanReportInfoVar	-
Buf-Scan-Report-Fixed	Подтвержденный или неподтвержденный	MDC_NOTI_BUF_SCAN_REPORT_FIXED	ScanReportInfoFixed	-

Окончание таблицы 18

Событие	Режим	Тип события	Event-info parameter	Event-reply-info
Buf-Scan-Report-Grouped	Подтвержденный или неподтвержденный	MDC_NOTI_BUF_SCAN_REPORT_GROUPED	ScanReportInfoGrouped	-
Buf-Scan-Report-MPVar	Подтвержденный или неподтвержденный	MDC_NOTI_BUF_SCAN_REPORT_MP_VAR	ScanReportInfoMPVar	-
Buf-Scan-Report-MPFixed	Подтвержденный или неподтвержденный	MDC_NOTI_BUF_SCAN_REPORT_MP_FIXED	ScanReportInfoMPFixed	-
Buf-Scan-Report-MPGrouped	Подтвержденный или неподтвержденный	MDC_NOTI_BUF_SCAN_REPORT_MP_GROUPED	ScanReportInfoMPGrouped	-

Все стили отчетов о событиях, перечисленных в таблице 18, являются буферизованными эквивалентами их небуферизованных двойников в 6.3.9.4.5. Первое различие заключается в том, что сканер буферизует данные в соответствии с интервалом отчетов и посылает единственное сообщение в конце интервала. Второе различие заключается в том, что в каждый отчет включены одни и те же объекты и атрибуты, независимо от того, изменились ли их значения.

6.3.9.5.6 Другие сервисы периодического конфигурируемого сканера

Нет.

6.4 Правила расширения информационной модели

Информационная модель расширяет свою сферу применения при помощи дополнительных атрибутов объектов для объектов, определенных в настоящем стандарте, определенных в ИСО/ИИЭР 11073-10201 [13].

Другим возможным расширением может стать использование частных (например, специфицированных производителем) атрибутов объектов и/или методов для объектов, определенных в настоящем стандарте. Частные атрибуты должны идентифицироваться путем присвоения номенклатурных кодов из частного пространства нумерации (0xF000 — 0xFFFF) в рамках соответствующего раздела в соответствии с ИСО/ИИЭР 11073-10101 [12].

Реализация системы менеджера должна полностью обрабатывать сообщение, пропуская любые неизвестные атрибуты (например, указанные продавцом атрибуты) и игнорируя присвоенные значения данных этих атрибутов без ошибок протокола. Реализация может внести соответствующую запись о таких атрибутах (например, в журнале регистрации).

7 Сервисная модель персонального медицинского прибора

7.1 Общие положения

Сервисная модель определяет концептуальный механизм для сервисов обмена данными. Данные сервисы отображаются в сообщениях, которыми обмениваются агент и менеджер. Протокольные сообщения в рамках серии стандартов ИСО/ИИЭР 11073 определены в языке ASN.1. Сообщения, описанные в настоящем стандарте, могут сосуществовать с сообщениями, описанными в других стандартных профилях, описанных в серии стандартов ИСО/ИИЭР 11073.

Сообщения протокола структурируются следующим образом:

- фрейм-структура протокола верхнего уровня отделяет сообщения команд, связанных с менеджментом связи (сообщения ассоциации), от сообщений верхнего уровня, связанных с объектом (передача данных и сервисов);
- фрейм-структура верхнего уровня, в частности, предоставляет тип сообщения и длину поля;
- протокол, при использовании правил MDER, позволяет агентам хранить заранее определенные шаблоны передачи и изменять только стабильное размещение, изменяя детали до отправки.

7.2 Сервис ассоциации

Сервис ассоциации управляет ассоциацией между агентом и менеджером. Частью сервиса ассоциации являются следующие сообщения:

- Запрос Ассоциации устанавливает соединение верхнего уровня над существующим транспортным соединением;
- Ответ на Ассоциацию принимает Запрос на Ассоциацию, если соединение двунаправленное;
- Запрос на Разъединение корректно завершает ассоциацию верхнего уровня;
- Ответ на Разъединение подтверждает завершение ассоциации верхнего уровня, если соединение двунаправленное;
- Отмена немедленно и без ответа останавливает верхний уровень ассоциации. Обычно это сообщение пересылается как результат ошибки.

7.3 Сервисы доступа к объектам

Сервисы доступа к объектам используются для доступа объектов информации, определенные в DIM. Эти сервисы, в частности, используются для сообщения данных и функций доступа к данным, предоставляемых агентом.

Поддерживаются следующие сервисы доступа обобщенного объекта:

- сервис GET: используется менеджером для извлечения значений объекта MDS агента и атрибутов PM-блока. Список атрибутов объекта системы MDS указан в 6.3.2.3, список атрибутов PM-блока указан в 6.3.7.3;
- сервис SET: используется менеджером для установки значений атрибутов объекта агента. Как правило, только объекты сканера поддерживают сервис SET (см. 6.3.9.2.6);
- сервис EVENT REPORT: используется агентом для пересылки обновлений конфигурации и измерительных данных менеджеру. Список отчетов о событиях указан в 6.3.2.5, 6.3.7.5, 6.3.9.4.5 и 6.3.9.5.5;
- сервис ACTION: используется менеджером для запуска действий (или методов), поддерживаемых агентом. Примером служит действие MDS-Data-Request, используемая для запроса измерительных данных у агента. Список методов указан в 6.3.2.4 и 6.3.7.4.

Доступ к объектам агента посредством запроса Get должен считаться неверным до тех пор, пока не выполняется одно из следующих условий:

- агент находится в рабочем состоянии и сервис GET ссылается на объект MDS или дескриптор объекта, который был заявлен в ходе конфигурации;
- агент находится в состоянии ассоциации и сервис GET ссылается на объект MDS.

Менеджер, получающий подтвержденный отчет о событиях от агента, должен ответить либо rors-cmipconfirmed-event-report, либо соответствующим сообщением об ошибке roer с подходящим обратным кодом.

Если запрос на подтвержденное действие получено агентом, который не поддерживает действие, агент должен ответить ошибкой (roer) со значением ошибки «не существующее действие». В случае ошибки выполнения подтвержденного действия, ошибка может быть указана, возвращая ошибку (roer) с соответствующим значением ошибки и, где применимо, может быть включена дополнительная информация об ошибке в поле параметра, используя один из обратных кодов из раздела обратных кодов.

7.4 Специальная реализация доступа к объекту сервисов EVENT REPORT для персональных медицинских приборов

7.4.1 Общие положения

Сервис EVENT REPORT является главным механизмом для агента посылающим отчеты как о данных измерений так и о данных конфигурации. Отчеты о событиях в настоящем стандарте являются собственностью системы MDS и объектов сканера. Такие особые отчеты о событиях могут иметь разные формы и свойства, в соответствии с 7.4.2 по 7.4.7.

7.4.2 Подтвержденные и неподтвержденные отчеты о событиях

Отправитель отчета о событиях может опционально запросить подтверждение от получателя.

7.4.3 Отчеты о событиях конфигураций

7.4.3.1 Общие положения

Подпункты 7.4.3.2—7.4.3.4 описывают конфигурации, отчеты о событиях конфигураций, и специализации прибора, используемые для описания объектов в агенте.

7.4.3.2 Конфигурация прибора-агента

Набор объектов и атрибутов, который существует в агенте, обозначает конфигурацию агента и ассоциируется со значением Dev-Configuration-Id (см. таблицу 2). В случае если агент обладает множественными конфигурациями прибора, присвоенные значения Dev-Configuration-Id должны быть локально-уникальными. В течение срока работы ассоциации, конфигурация агента должна оставаться стабильной, то есть набор объектов должен оставаться неизменным. Однако агент может добавить новые атрибуты объекту или изменить значения атрибута в соответствии с 7.4.3.3. Агент, который запрашивает другую конфигурацию, должен остановить ассоциацию и установить новую ассоциацию с желаемой конфигурацией.

7.4.3.3 Отчет о событиях конфигураций

Конфигурация, которую агент желает использовать во время ассоциации с менеджером, указывается при помощи значения Dev-Configuration-Id для поля dev-config-id в сообщении Запрос ассоциации. Если менеджер еще не знаком с конфигурацией прибора-агента (например, на основе предыдущих фаз ассоциации), менеджер запрашивает конфигурацию прибора агента. Агент направляет свои конфигурации менеджеру при помощи отчета о событиях конфигураций. Отчет описывает все объекты конфигурации прибора агента вместе с ассоциированным значением Dev-Configuration-Id. В течение ассоциации, конфигурации агента остаются стабильными в отношении количества объектов. В случае если агент намеревается использовать другую конфигурацию или желает изменить текущую конфигурацию путем добавления или удаления объектов, агент должен прервать ассоциацию и произвести новую ассоциацию с новой конфигурацией. Для каждого объекта конфигурация должна содержать также атрибуты, используемые данным объектом. Обычно, редко изменяемые атрибуты включаются в отчет конфигурации, а динамические значения как, например, измерения, пересылаются в последующих отчетах об измерениях. Агент может добавлять новые атрибуты объекту или изменять значения атрибутов в ходе ассоциации без пересылки новой конфигурации. Добавление новых атрибутов может случиться только при помощи отчета о событиях переменного формата (см. 7.4.5 для подробной информации о форматах отчетов о событиях). Изменяемые значения атрибутов могут использовать переменные, фиксированные или групповые отчеты о событиях в зависимости от конфигурации.

Изменения, вносимые в текущую конфигурацию, как стандартную, так и расширенную, имеют силу только в течение той ассоциации, и не считаются устойчивым изменением конфигурации. Следовательно, Dev-Configuration-Id представляет собой конфигурацию, согласованную на время конфигурации. В последующих ассоциациях, когда указывается ранее используемый Dev-Configuration-Id, текущая конфигурация не включает каких-либо изменений, внесенных в ходе предыдущей ассоциации. Устойчивые изменения конфигурации могут быть внесены только после установления новой ассоциации и указания другого Dev-Configuration-Id, и желаемой новой конфигурации во время настройки.

Менеджер использует информацию о конфигурации для создания эквивалентной модели информации агента. Затем данная информация обновляется агентом в качестве собранных измерений.

7.4.3.4 Специализации прибора

Специализации прибора ИСО/ИИЭР 11073-104zz определяют обязательные атрибуты и объекты, которые должны существовать в рамках конфигурации агента. Более того, каждая из специализаций определяет обязательные элементы (например, включающие обязательные действия и методы) моделей сервиса и связи, которые должны поддерживаться агентом в соответствии с этой специализацией.

7.4.3.5 Типы конфигурации

Для уменьшения размеров сообщения передачи, настоящий стандарт устанавливает способность информировать менеджера о конфигурациях агента эффективным образом. Существует два типа конфигураций: стандартная и расширенная.

7.4.3.5.1 Стандартная конфигурация

Стандартная конфигурация — это конфигурация, оговоренная в одной из специализации ИСО/ИИЭР 11073-104zz и со значением Dev-Configuration-Id, присвоенным из диапазона между standard-config-start и standardconfig-end, включительно. Этот диапазон ниже подразделен, резервируя 100 ID на каждую специализацию ИСО/ИИЭР 11073-104zz в диапазоне от zz × 100 до zz × 100 + 99, включительно. Например, диапазон 1500—1599 зарезервирован под ИИЭР Std 11073-10415 [B4]. Все

неиспользуемые значения в стандартном диапазоне зарезервированы под будущее использование. Менеджер, который поддерживает одну из специализаций прибора ИСО/ИИЭР 11073-104zz, знаком со всеми конфигурациями стандартных приборов, указанных в конкретной специализации. Каждый раз, как агент запрашивает ассоциацию с менеджером при помощи значения Dev-Configuration-Id стандартной конфигурации, менеджер может принять ассоциацию без необходимости запроса конфигураций агента, т. к. они уже известны. После успешной ассоциации менеджер и агент входят в режим Работы.

Важно отметить, что при наличии запроса приборы со стандартными конфигурациями должны отправлять свои конфигурации. Это требование покрывает случай, когда агент ассоциируется с менеджером, в который не была встроена информация о стандартных конфигурациях (например, версия менеджера 1.0, а специализация прибора — версия 2.0 и выше). Способность менеджера применять конфигурации зависит от реализации менеджера.

Если агент использует значение Dev-Configuration-Id, присвоенное стандартной конфигурации, он должен также заполнить все дополнительные обязательные элементы (например, включая обязательные действия и методы) моделей сервиса и связи, в соответствии со специализацией прибора.

7.4.3.5.2 Расширенная конфигурация

При расширенных конфигурациях, конфигурация агента не является стандартной; он может иметь различный набор объектов, различные атрибуты и/или различные значения атрибутов. Расширенная конфигурация(-ии) применения агента должна выбрать уникальное значение Dev-Configuration-Id из диапазона между extended-configstart и extended-config-end, включительно для каждой расширенной конфигурации. За время ассоциации агент отправляет Dev-Configuration-Id для определения конфигурации, выбранной агентом, на период проведения ассоциации. Если менеджер уже понимает эту конфигурацию, либо потому, что она была загружена ранее при помощи программы установки, либо потому, что агент ранее устанавливал ассоциацию с менеджером, то менеджер должен ответить принятием конфигурации без необходимости пересылки дальнейшей информации о конфигурациях.

Однако, если менеджер не знаком с конфигурацией агента, менеджер должен ответить реакцией accepted-unknown-config, после чего агент должен передать информацию о своих конфигурациях, отправляя отчет о событии конфигурации. См. 8.7 и 8.8 для подробной информации о процедурах ассоциации и конфигурации. Как только менеджер получает конфигурацию, агент может передать измерительные данные. Для экономии времени ассоциации, агентом должен использоваться тот же Dev-Configuration-Id для последующих ассоциаций при условии неизменности конфигурации прибора.

В отличие от стандартных конфигураций, два агента с одинаковым расширенным Dev-Configuration-Id не обязательно представляют одну и ту же конфигурацию. Менеджер должен различать расширенные конфигурации на базе каждого агента. System-Id агента может использоваться для различия расширенных конфигураций, т. к. System-Id является обязательной и должна быть уникальной, и высылается во время ассоциации; однако вместо них могут использоваться другие техники, как например, производитель/модель/серийный номер, только если они не вынуждают менеджера использовать некорректную конфигурацию для агента.

По существу, агент с расширенной конфигурацией поддерживает ноль, одной или несколько специализаций прибора в соответствии со спецификациями ИСО/ИИЭР 11073-104zz. В случае, когда агент поддерживает одну и более специализации прибора, он должен применять все обязательные и действующий набор условных пунктов (включая объекты, атрибуты, действия и методы), указанных в соответствующих спецификациях.

7.4.4 Передача измерительных данных, инициируемая агентом или менеджером

Передача измерительных данных, инициируемых агентом, выполняется агентом, например, как результат сделанных новых измерений.

Передача измерительных данных, инициируемая менеджером, напрямую запрашивается менеджером путем выпуска команды (т. е. MDS-Data-Request) для сообщения агенту информации о начале и окончании пересылки измерительных данных. В зависимости от мощности агента, настраивается отрезок времени, в течение которого остается активным этот режим отчета (например, установленный период или всегда в ассоциации).

Менеджер должен поддерживать получение Передачи измерительных данных, инициируемой агентом или менеджером от агента. Агент может поддерживать генераются передачи измерительных данных, инициируемую одним из двух или обоими участниками обмена.

В обоих случаях Передачи измерительных данных, используются отчеты о событиях для обработки измерительных данных.

7.4.5 Переменный, фиксированный и групповой форматы отчетов о событиях

Сообщение о событии может протекать в трех стилях: переменный, фиксированный и групповой форматы. Рисунок 7 демонстрирует взаимосвязь между каждым из форматов сообщений.

Переменный формат отчета о событиях прямо определяет каждый сообщаемый атрибут, включая поле идентификации атрибута, длину значения и значение в сообщении. Данный подход обеспечивает гибкость во включении различных наборов атрибутов на отчет о событиях, но за счет расходов сообщения.

Фиксированный формат отчета о событиях оптимизирован, за счет определения формата сообщения в карте Attribute-Value-Map объекта в предыдущих сообщениях конфигурации до начала пересылки. Когда агент передает данные в фиксированном формате отчета о событиях, он должен сообщать дескриптор объекта и значения атрибута в порядке и размере, указанном в карте значений атрибута Attribute-Value-Map. Таким образом, предотвращается расход операции при отправке идентификации и длины атрибута в каждом отчете о событиях. По получении отчета о событиях в фиксированном формате, менеджер использует дескриптор объекта для извлечения карты Attribute-Value-Map, выданной ранее во время конфигурации, чтобы получить информацию о способе извлечения данных.

Групповой формат отчета о событиях оптимизирован дополнительно за счет определения формата сообщения, содержащего один и более объектов, в объекте сканера Handle-Attr-Val-Map в отдельном сообщении конфигурации до начала передачи. Когда агент передает данные в групповом фиксированном формате отчета о событиях, он должен сообщать дескриптор объекта сканера вместе со значениями атрибута объектов в порядке и размере, указанном в карте значений атрибута Handle-Attr-Val-Map. Таким образом, предотвращаются издержки операции при отправке дескрипторов объекта сканера, идентификации атрибута и длины данных в каждом отчете о событиях. По получении отчета о событиях в групповом формате, менеджер использует дескриптор объекта сканера для извлечения карты the Handle-Attr-Val-Map, выданной ранее во время конфигурации, чтобы получить информацию о способе извлечения данных.

Менеджер должен поддерживать переменный формат, фиксированный формат и групповой формат отчетов о событиях. Агент может поддерживать любой или все отчеты о событиях с переменным форматом, фиксированным форматом и групповым форматом. Менеджер определяет, какой из форматов может использовать агент, проверяя карты объектов Attribute-Value-Map, определенной в MDSConfiguration-Event от агента или проверяя атрибут Handle-Attr-Val-Map на объекты сканера, определенных в MDS-Configuration-Event.

7.4.6 Отчеты о событиях одного и нескольких лиц

Агенты, созданные для работы в среде, где данные могут собираться от нескольких лиц, могут использовать отчет о событиях для нескольких лиц, чтобы передавать все данные от всех лиц в одном событии. Там, где эта функция не требуется, агент может использовать отчет о событиях для одного лица для снижения расходов.

Менеджер должен поддерживать отчеты о событиях как для одного, так и нескольких лиц. Агент может поддерживать, как только один из этих вариантов, так и оба варианта отчета о событиях для одного и нескольких лиц. В А.11.5 описаны форматы для отчетов о событиях одного или нескольких лиц.

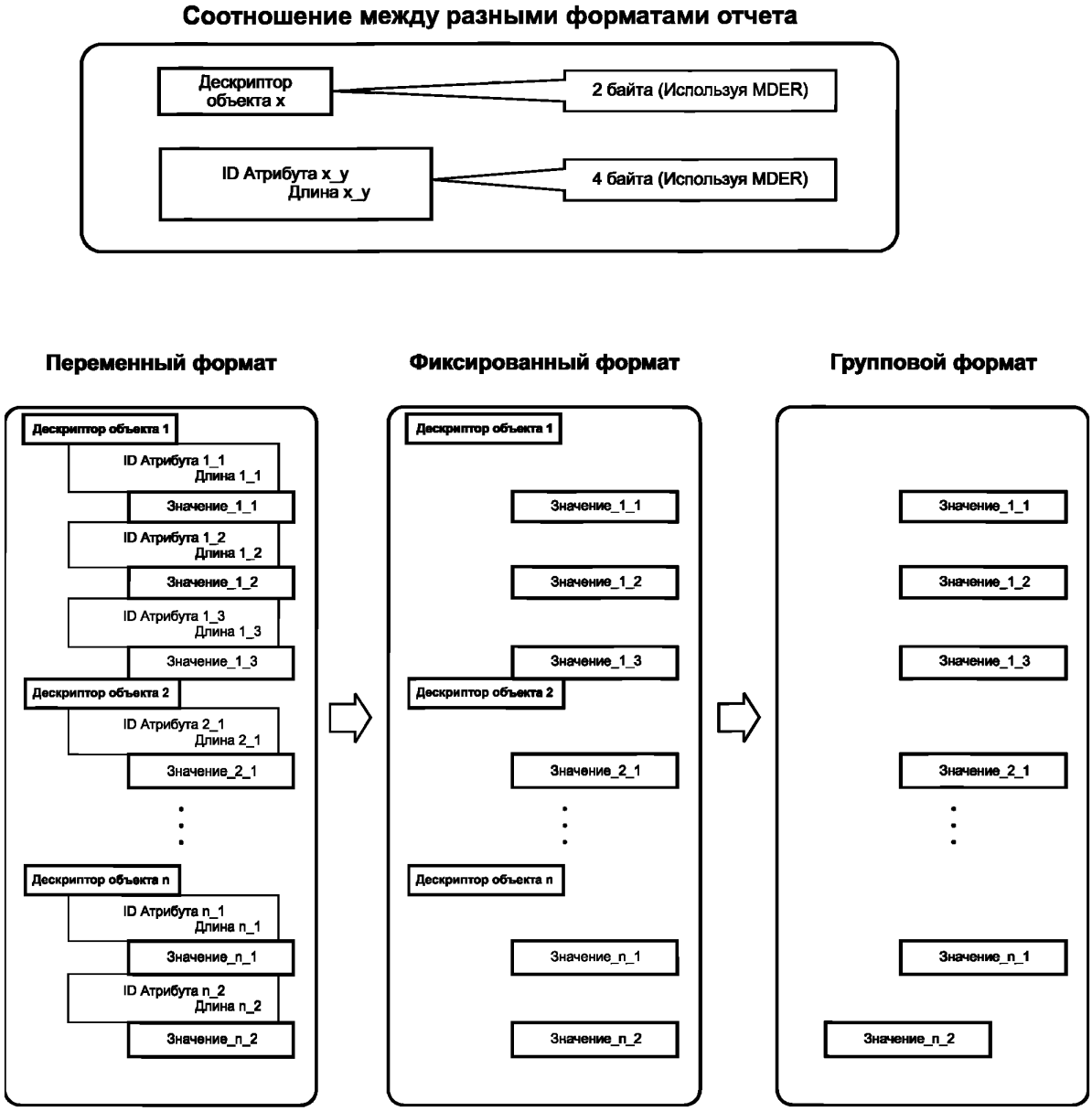


Рисунок 7 — Связи переменного, фиксированного и группового формата

7.4.7 Временно хранимые измерения

Агент может на выбор хранить небольшое количество измерений в локальной памяти, в то время как он не соединен с системой менеджера (т.е. временно хранимые измерения). Как только агент может установить соединение с менеджером, все ранее сохраненные измерения передаются менеджеру.

Примечание — Типичным примером временно-хранимых измерений являются весы: новые измерения производятся редко. Весы не соединены с менеджером, и они отключаются после проведения измерения вместо того, чтобы ждать неопределенное время менеджера и потребления энергии.

Для поддержки временно хранимых измерений, система агента должна применять следующее поведение:

- только объекты с метрическим происхождением, не являющиеся RT-SAs (например, числовые объекты и объекты перечисления), поддерживаются как временно хранимые измерения;
- использование атрибутов временных отрезков (т. е. Date-and-Time, Relative-Time или HiRes-Relative-Time) требуется для временно хранимых измерений;

- агент не должен пересылать временно хранимые измерения, если известно, что временной отрезок не соблюдается (например, если временная база, используемая для отсчета временного отрезка значений, притерпела между измерениями изменения значительные для типа измерения), если только он не включает соответствующие настройки Date-and-Time-Adjustment в начале отчета о событии;

- временно хранимые измерения включены в любой из предписанных механизмов отчета о событии (инициируемые агентом или менеджером; групповой, фиксированный или переменный форматы; одного или нескольких лиц);

- после передачи временно хранимых измерений менеджеру, агент должен удалить хранимые измерения из локальной памяти. Агент должен убедиться, что измерения успешно переданы менеджеру при помощи подтвержденных отчетов о событиях;

- для ограничения количества данных, передаваемых данным механизмом, агент должен обеспечить не более 25 временно хранимыми измерениями в любом отчете о событиях. Если требуется хранение более 25 измерений, необходимо использовать механизм PM-блока для архивирования измерений.

8 Модель коммуникаций

8.1 Общие положения

В целом, ожидаемой топологией является процесс, при котором один или несколько агентов производят обмен информацией посредством двухточечного соединения с менеджером. Если менеджер хочет поддерживать множество агентов одновременно (например, используя пикосеть Bluetooth), то он должен иметь возможность обрабатывать множество индикаторов соединения и отдельные ассоциации от каждого из этих агентов.

Любой агент, поддерживающий несколько модальностей (специализаций устройств), может выбрать создание одного подключения и ассоциации с менеджером или создать несколько индикаций соединений и ассоциаций (например, один для каждой модальности) с менеджером. Однако если агент выбирает создание нескольких индикаций соединений и ассоциаций, экземпляры объектов в различных ассоциациях должны быть полностью независимыми, как если бы ассоциации были созданы различными устройствами. В качестве примера объект MDS для каждой индикации соединения и ассоциации должен действовать в качестве отдельных независимых агентов.

8.2 Контекст системы

Коммуникационный профиль, определенный в настоящем стандарте учитывает специфические требования агентов и менеджеров персональных медицинских приборов, которые обычно используются в подвижной среде или доме человека. Следующие допущения приняты в отношении сервисов и свойств, которые должны быть предоставлены транспортным уровнем. Кроме того, также охвачен контекст системы за пределами этого коммуникационного профиля (то есть не персональные медицинские приборы/приборы для поддержания функциональности прикладного уровня) и его связь с допущениями транспортных уровней.

Настоящий стандарт предполагает, что транспортные технологии обладают широким спектром свойств, и в общих чертах рассматривает транспортные технологии для того, чтобы позволить использование и применение в исходном формате их сервисов. Если передача данных не программируема в исходном формате, для удовлетворения требуемых характеристик добавляется «промежуточное средство».



Рисунок 8 — Контекст системы

Настоящий стандарт использует понятие «тип» для группировки и дифференциации сервисов, предлагаемых доступными транспортными технологиями, которые были профилированы для использования серией стандартов ИСО/ИИЭР 11073. В частности, серия стандартов ИСО/ИИЭР 11073 распознает следующие типы транспортных профилей:

- Тип 1. Транспортные профили, содержащие «надежные» и «лучшие из возможных» транспортные сервисы, в которых должен быть один или несколько виртуальных каналов «надежных» транспортных сервисов и ноль или более виртуальных каналов «лучших из возможных» транспортных сервисов;
- Тип 2. Транспортные профили, содержащие только однонаправленные транспортные сервисы;
- Тип 3. Транспортные профили, содержащие только «лучшие из возможных» транспортные сервисы, в которых должен быть один или несколько виртуальных каналов «лучших из возможных» транспортных сервисов.

Причина, по которой типы транспортных профилей являются значимыми, заключается в том, что различные транспортные сервисы, предлагаемые типами транспортных профилей, оказывают влияние на осуществление некоторых функций верхнего уровня. В частности, они оказывают влияние на осуществление подтвержденного этим стандартом сервисного механизма. Настоящий стандарт предназначен для использования только с транспортными профилями 1-го типа.

Более полное описание различных транспортных типов профиля и как они взаимодействуют с подтвержденными и неподтвержденными механизмами службы см. в приложении D.

8.3 Коммуникационные характеристики

8.3.1 Общие положения

В настоящем стандарте должен быть использован транспортный профиль 1-го типа.

В настоящем стандарте каждое устройство должно поддерживать основной виртуальный канал. Основной виртуальный канал должен являться надежным виртуальным каналом (то есть надежным транспортным сервисом) из транспортного профиля 1-го типа.

Основной виртуальный канал должен использоваться для следующих целей:

- все сообщения, относящиеся к процедуре ассоциации:
  - aare, aarq, rlre, rlrq, abrt;
- все сообщения, относящиеся к утвержденному сервисному механизму:
  - prst.roiv-cmip-confirmed-action, prst.roiv-cmip-confirmed-event-report, prst.roiv-cmip-get, prst.roiv-cmip-confirmed-set;
  - prst.rors-cmip-confirmed-action, prst.rors-cmip-confirmed-event-report, prst.rors-cmip-get, prst.rors-cmip-confirmed-set;
- все сообщения, относящиеся к условиям сбоя, аварийным условиям:
  - roer, rorj.

В настоящем стандарте каждое устройство может поддерживать один или несколько вторичных виртуальных каналов. Каждый вторичный виртуальный канал может являться либо надежным, либо лучшим из возможных из транспортного профиля 1-го типа.

Основной виртуальный канал или любой вторичный канал(ы) может быть использован для передачи сообщений, связанных с неподтвержденным сервисным механизмом.

- prst.roiv-cmip-action, prst.roiv-cmip-event-report, prst.roiv-cmip-set

Как правило, термин метаданные означает данные о данных. В контексте коммуникационных характеристик ИСО/ИИЭР 11073-20601, термин метаданные означает подтверждающую информацию или данные, относящиеся к пакету данных протокола прикладного уровня (APDU). Примеры включают следующее:

- специфический транспортно-технологический адрес для передачи данного пакета APDU данному агенту или менеджеру;
- надежность и/или скрытость, необходимая для данного пакета APDU;
- размер или длина данного пакета APDU.

Некоторые метаданные описывают коммуникационные характеристики, представленные как отдельное значение, которое охватывает широкий диапазон возможных значений. Что касается общих примеров метаданных, приведенных выше, некоторые конкретные примеры имеют следующий вид:

- APDU-metadata.address (для конечной точки USB) = 1 – 1023;

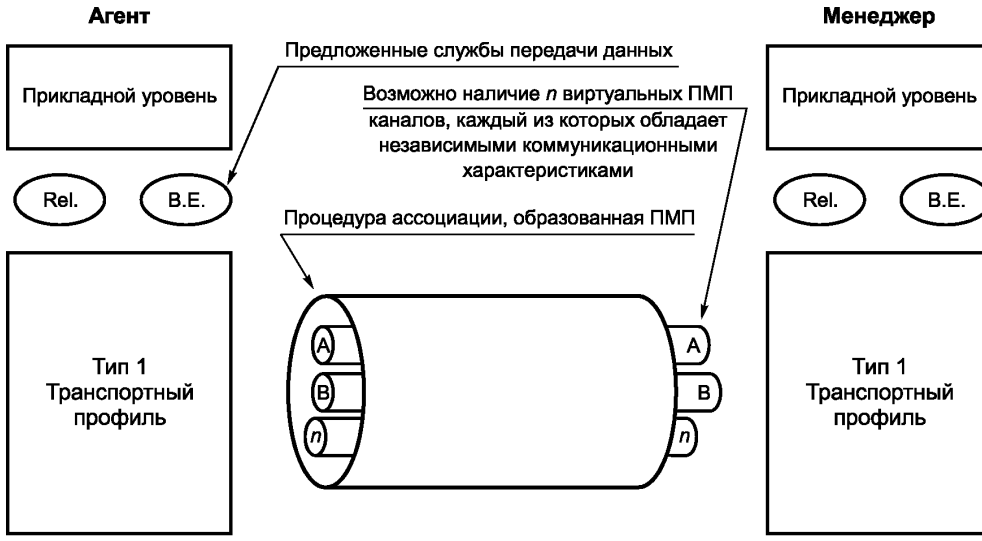


Рисунок 9 — Общая модель коммуникации

- APDU-metadata.address (для сети IPv4) = 0.0.0.0 – 255.255.255.255;
- APDU-metadata.size = 1 – 64512.

Другие метаданные описывают коммуникационные характеристики, представленные как отдельное значение, но обладающие только несколькими дискретными, возможными значениями. Что касается общих примеров метаданных, приведенных выше, некоторые конкретные примеры имеют следующий вид:

- APDU-metadata.latency = (10ms | 100ms | 1sec | 10sec);
- APDU-metadata.reliability = (high | medium | low);
- APDU-metadata.bandwidth = (100 бит/с | 1 Кбит/с | 10 Кбит/с | 100 Кбит/с | 1 Мбит/с).

В последующих подразделах описываются общие характеристики (см. 8.3.2) и уникальные характеристики надежных (см. 8.3.3) лучших из возможных (см. 8.3.4) виртуальных каналов применительно к настоящему стандарту.

8.3.2 Общие коммуникационные характеристики

Ряд общих коммуникационных характеристик, применимых для надежных и лучших из возможных коммуникаций:

a) пакет данных APDU может быть обработан любым способом (например, часть за частью, как полученный APDU или полный APDU буферизированный в памяти), однако пакет данных APDU должен быть обработан таким образом, чтобы эффект от него был такой же как от атомной транзакции;

b) пакеты данных APDU могут быть сегментированы и собраны повторно во время передачи, или он может быть отправлен как единое целое;

c) пакеты данных APDU, отправляемые от агента к менеджеру, должны составлять не более 63 Кбайт (64 512). Особые специализации устройств или реализации могут оценить сообщения, переданные с целью определения конкретного размера реализации для приемного буфера менеджера, меньшего, чем максимальный размер пакета APDU, переданного от агента к менеджеру. Если менеджер получает больший по размеру пакет данных APDU, то он должен ответить, указав код ошибки (Roer) из протокола-нарушения;

d) пакеты данных APDU, отправляемые от менеджера к агенту, должны составлять не более 8 Кбайт (8192). Особые специализации устройств или реализации могут анализировать сообщения, которыми обмениваются с целью определения конкретного размера реализации для приемного буфера агента, меньшего, чем максимальный размер пакета APDU, передаваемый от менеджера к агенту. Если

агент получает больший пакет данных APDU, то он должен ответить, указав код ошибки (Roer) из протокола-нарушения;

e) общая длина пакета данных APDU должна поступать на и из коммуникационных уровней как метаданные;

f) коммуникационные уровни должны указывать общую длину пакета данных APDU аналогичному коммуникационному уровню.

8.3.3 Характеристика надежных коммуникаций

К коммуникационным технологиям/методам, которые должны рассматриваться как надежные, пригодные для использования оптимизированным протоколом обмена, относятся следующие характеристики:

a) пакеты данных APDU должны быть получены в том же порядке, в котором были отправлены;

b) пакеты данных APDU не должны содержать обнаруживаемых ошибок;

c) Пакеты данных APDU не должны копироваться;

d) Пакеты данных APDU не должны быть утеряны;

e) пакеты данных APDU, как правило, отправляются в срочном порядке, однако допустимы задержки в связи с повторными передачами;

f) коммуникационный уровень должен предоставлять механизм для указания прикладного уровня, при получении полного пакета APDU;

g) коммуникационный уровень должен предоставлять механизм для указания прикладного уровня, при установлении путей связи между агентом и менеджером;

h) коммуникационный уровень должен предоставлять механизм для указания прикладного уровня, когда соединение прекращается или прерывается;

i) коммуникационный уровень должен предоставлять механизм для указания прикладного уровня, если отправка APDU невозможна;

j) управление потоком между отправляющим и получающим приложением должно поддерживаться для полного пакета APDU. Нижние слои могут реализовать управление потоком для небольших подклассов APDU.

8.3.4 Характеристика лучших из возможных коммуникаций

Если коммуникационные технологии не соответствуют критерию надежного канала коммуникации, согласно описанию выше, они именуются оптимизированным протоколом обмена как лучшие из возможных. Следующие характеристики типичны для канала лучшего из возможных:

a) пакет APDU не может быть доставлен в том порядке, в котором он был отправлен. Сам коммуникационный канал, независимо от работы передатчика персонального медицинского прибора, может нарушить порядок пакетов;

b) пакет APDU может быть утерян или скопирован;

c) пакеты APDU могут поступить в размере, который приводит к опустошению буфера-получателя.

8.4 Конечные автоматы

8.4.1 Конечный автомат агента

На рисунке 10 представлен общий вид конечного автомато агента.

Подробная таблица состояний агента описана в приложении Е.2.

В таблице 19 дано описание каждого состояния.

В таблице 20 дано описание каждого изменения состояния.

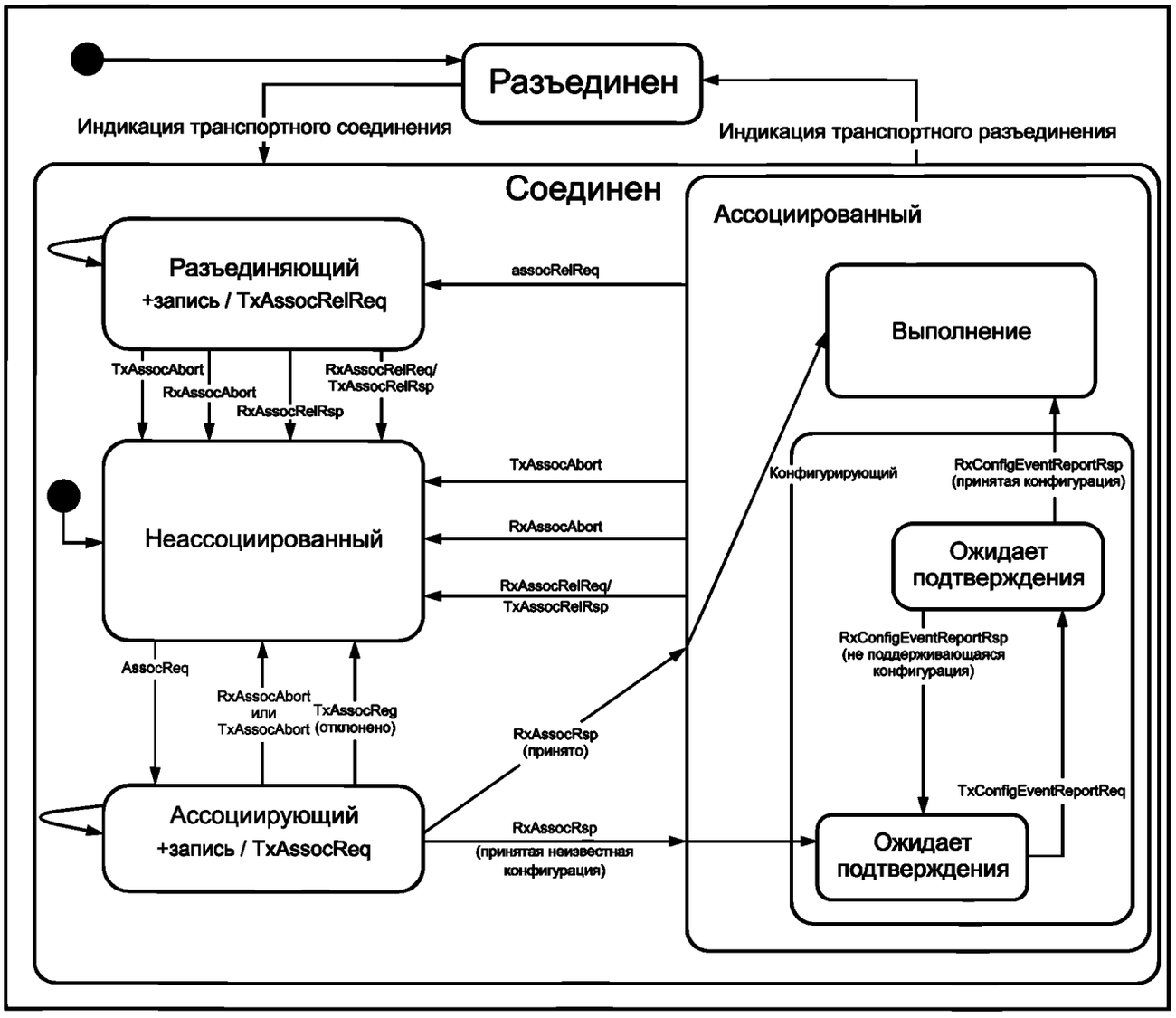


Рисунок 10 — Диаграмма конечного автомата агента

Таблица 19 — Описание состояний агента

Состояние	Описание
Разъединен	Если изначально агент включается, это допускается при начале работы в отключенном состоянии, которое указывает на то, что транспортное соединение между агентом и менеджером не было создано. После того как транспортное соединение будет установлено, можно вернуться в отключенное состояние, если транспортное соединение было намеренно завершено или непреднамеренно отключено
Соединен	Когда транспортное соединение установлено, агент получает индикацию транспортного соединения из транспортного уровня, вызывая переход в состояние соединения (см. 8.4.3). Агент остается в состоянии соединения, пока транспортное соединение установлено. Изначально агент начинает работу в неассоциированном состоянии, в подсостоянии состояния соединения
Неассоциированный	Агент находится в неассоциированном состоянии, когда он не имеет ассоциацию прикладного уровня с менеджером. Такая ситуация может возникнуть в связи с любым из следующих условий: - новое соединение было только что установлено; - менеджер отвергает запрос на ассоциацию; - любая из сторон завершает или разрывает активную ассоциацию в любой момент при подключении. Агент остается в неассоциированном состоянии, пока не определит, что он должен начать ассоциироваться с менеджером

Окончание таблицы 19

Состояние	Описание
Ассоциирование	Всегда, когда агент определяет, что должен создать ассоциацию, агент переходит в состояние ассоциирования и посылает запрос на ассоциацию менеджеру (см. п. 8.6). Если ассоциацию не удается создать, но альтернативные параметры ассоциации возможны, агент может попытаться связаться с помощью каждого нового набора параметров ассоциации. В случае тайм-аута, агент должен продолжать попытку связаться до тех пор, пока максимальное число повторных попыток не будет достигнуто или ассоциация не будет создана
Ассоциированный	Когда менеджер определяет, что агент и менеджер используют общие версии и протоколы, он посылает ответ на ассоциацию с параметром «accepted» (см. 8.7.3.3) агенту. Когда агент получает это сообщение, он переходит в ассоциированное состояние. Он остается в этом состоянии до тех пор, пока агент не отправит или получит запрос на выпуск или разрыв ассоциации. Первоначальное подсостояние при входе в ассоциированное состояние зависит от того, ответит ли менеджер на запрос на ассоциацию, указав, признана ли конфигурация агента
Выполнение	Когда менеджер признает конфигурацию агента, он информирует его, посылая ответ на ассоциацию с параметром «accepted», чтобы перевести агента в рабочее состояние. В ином случае, если конфигурация не признана, она передается. Если конфигурация будет принята, агент переходит в состояние выполнения. См. 8.9 для получения описания возможных процедур в состояние выполнения
Конфигурирующее	Когда менеджер не признает конфигурацию агента, он информирует его, посылая ответ на ассоциацию с параметром «accepted-unknown-config», с целью указать, что ассоциация была принята, но, что конфигурация должна быть передана. Агент остается в конфигурирующем состоянии до тех пор, пока агент не передаст информацию о конфигурации и менеджер признает конфигурацию (см. 8.7.6)
Завершение ассоциации	Всегда, когда агент определяет, что он должен завершить текущую ассоциацию, агент переходит в состояние завершения ассоциации и посылает запрос на завершение ассоциации с менеджером (см. 8.10). В случае тайм-аута агент посылает запрос на завершение ассоциации и переходит в неассоциированное состояние

Таблица 20 — Описание изменения состояний агента

Состояние	Описание
Индикация транспортного соединения	Передача индикации транспортного соединения происходит всякий раз, когда средство передачи (или поддерживающий прокладочный уровень) указывает на то, что соединение установлено
assocReq	Каждый раз, когда агент определяет, что хочет попытаться создать ассоциацию с менеджером, он переходит в ассоциирующее состояние
RxAssocRsp (принята или принята-неизвестная-конфигурация)	Так как агент пытается создать ассоциацию с менеджером, он посылает сообщение с запросом на ассоциацию (или несколько в условиях тайм-аута) и ждет ответ на ассоциацию от менеджера. При получении агентом одобрения ассоциации, он переходит в соответствующее состояние
RxAssocRsp (отклонено)	Если менеджер определяет, что он не в состоянии связаться с агентом после получения запроса на ассоциацию, он посылает ответ на ассоциацию с кодом результата ассоциации (AssociateResult) отклонен, как постоянно или временно, чтобы перевести агента обратно в неассоциированное состояние

Окончание таблицы 20

Состояние	Описание
RxAssocRelReq/ TxAssocRelRsp	Если агент связан с менеджером и получает запрос на выпуск ассоциации, агент отвечает и переходит в состояние разъединения
RxAssocAbort илиTxAssocAbort	Всегда, когда агент и менеджер создают ассоциацию, ассоциированы или разъединяются, агент может или отправить, или получить сообщение о разрыве ассоциации. При возникновении данной ситуации, агент переходит из текущего состояния в неассоциированное состояние. Если агент ассоциирован, он может отправить сообщение о разрыве ассоциации для того, чтобы сообщить менеджеру, что произошел серьезный сбой. Это сообщение должно быть последним средством с предпочтением к отправке запроса на выпуск ассоциации, с целью корректно перейти в неассоциированное состояние. Если агент получает сообщение о разрыве ассоциации, для него нет необходимости отвечать, так как это сообщение может быть получено, только в случае, если ассоциацию разрывает менеджер (например, при аварии)
assocRelReq	Когда агент решает прекратить ассоциацию, он переходит в состояние разъединения и посылает запрос на выпуск ассоциации. Этот переход используется при обычной последовательности отключения путем отправки ReleaseRequestReason при обычных условиях, или, если конфигурация агента изменилась и ему необходимо прервать ассоциацию, агент использует ReleaseRequestReason об изменении конфигурации. В любом случае, в следующий раз, когда агент будет ассоциироваться, конфигурация, необходимая для использования в запросе на ассоциацию отображается, и менеджер определяет, знакома ли эта конфигурация
RxAssocRelRsp	Этот переход указывает на то, что запрос на выпуск текущей ассоциации был принят. Случай, когда агент послал запрос на разъединение ассоциации, означает, что агент получил ответ на выпуск ассоциации, указывающий на то, что завершение одобрено менеджером
Индикация разъединения передачи данных	В любой момент агент или менеджер может прекратить транспортное соединение или соединение может быть потеряно вследствие аварийных условий. Когда индикация того, что передача данных была отключена, получена, агент переходит в состояние отключения

8.4.2 Конечный автомат менеджера

Общий вид конечного автомата менеджера изображен на рисунке 11. Большинство состояний и переходов симметричны позициям, описанным для агента в таблице 19 и таблице 20. Основные различия заключаются в следующем:

- менеджер должен ждать в состоянии ожидания конфигурации как минимум TO_{config} (время выполнения процедуры конфигурации) секунд перед получением запроса на разъединение ассоциации или сообщения о разрыве ассоциации;

- если менеджер не принимает конфигурацию, он должен послать ответ на конфигурацию с результатом — unsupported-config;

- если менеджер принимает конфигурацию, он должен отправить ответ на конфигурацию с указанием результата — accepted-config.

Подробная таблица о состояниях менеджера приведена в приложении Е.3.

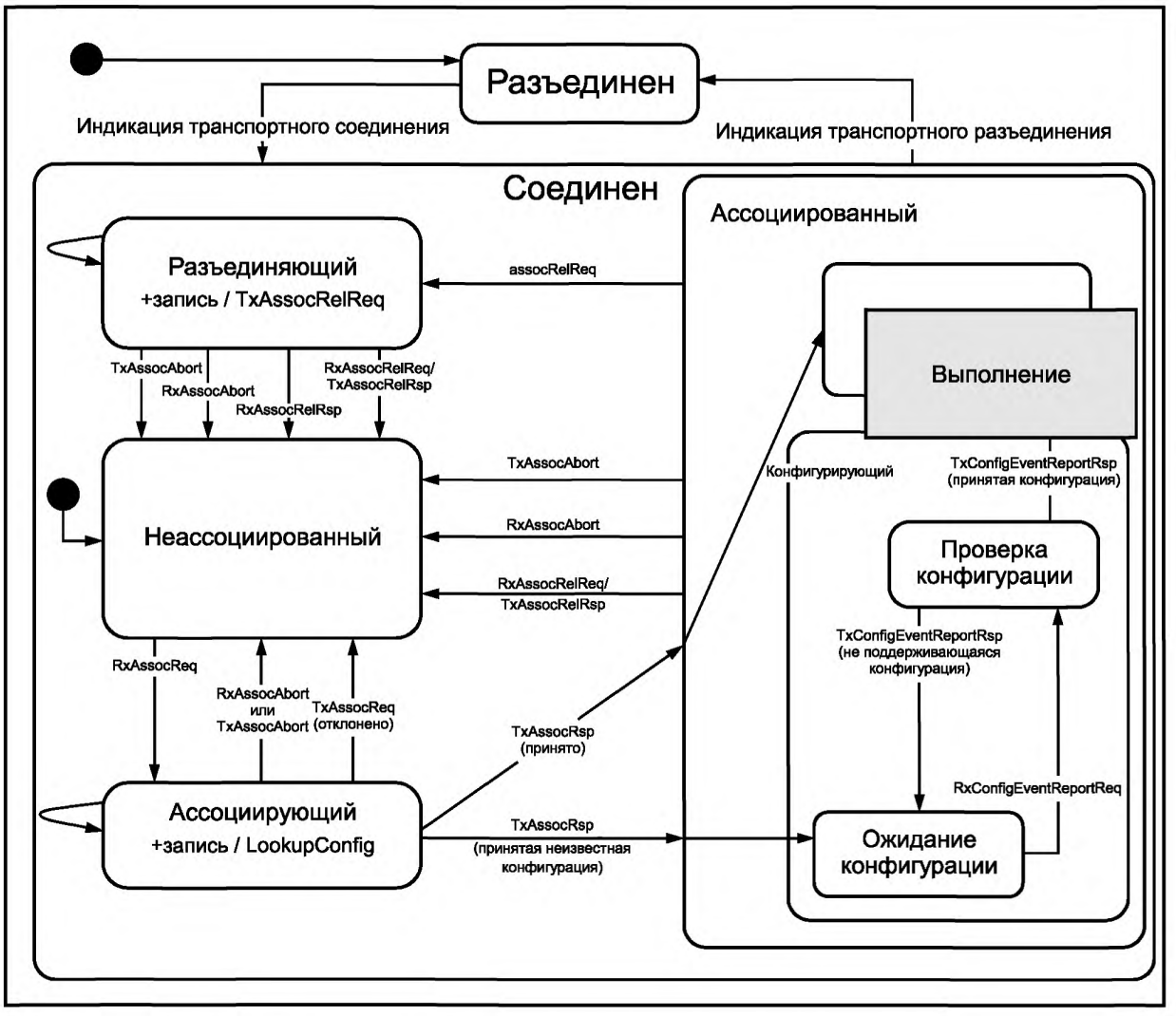


Рисунок 11 — Диаграмма конечного автомата менеджера

8.4.3 Переменные тайм-аута

В протоколе персонального медицинского прибора есть несколько разделов, где используются тайм-ауты. Имеются как тайм-ауты (периоды ожидания) для повтора, так и счетчики повторов. Чтобы облегчить управление документами в течение долгого периода времени и упростить электронный «поиск» для разных значений тайм-аута, конкретные численные значения были вынесены из содержания настоящего стандарта и заменены точными переменными тайм-аута. Численные значения тайм-аутов содержатся в таблице 21.

Таблица 21 — Переменные тайм-аута

	Коммуникационные сервисы	Тайм-аут		Подраздел
		Переменная	Значение	
Процедура ассоциации				
	Ассоциация	TO_{assoc}	10с (и RC_{assoc} = 3)	8.7.5
	Конфигурация	TO_{config}	10с	8.8.5
	Выпуск ассоциации	$TO_{release}$	3с	8.10.5

Окончание таблицы 21

	Коммуникационные сервисы	Тайм-аут		Подраздел
		Переменная	Значение	
Процедура работы				
Объект системы мед. прибора (MDS)	Подтвержденное действие	TO_{ca}	3с	8.9.5.2
	Подтвержденный отчет о событии	$TO_{cer-mds}$	Подтвержденный тайм-аут MDS	8.9.5.3
	Получение	TO_{get}	3с	8.9.5.4
	Подтвержденная установка	TO_{cs}	3с	8.9.5.5
	<inter-service timeout>	TO_{sp-mds}	3с	8.9.5.6
Объекты хранения РМ	Подтвержденное действие	TO_{ca}	3с	8.9.5.2
	Подтвержденный отчет о событии	$TO_{cer-pms}$	Подтвержденный тайм-аут сегмента	8.9.5.3
	Получение	TO_{get}	3с	8.9.5.4
	Подтвержденная установка	TO_{cs}	3с	8.9.5.5
	<end of Segm timeout>	TO_{sp-pms}	Тайм-аут передачи сегмента	8.9.5.6
	Подтвержденное действие — SegmClear	$TO_{clr-pms}$	Тайм-аут очистки сегмента	8.9.5.6
Объект сканера	Подтвержденная установка	TO_{cs}	3с	8.9.5.5
	Подтвержденный отчет о событии	$TO_{cer-scan}$	Подтвержденный тайм-аут сканера	8.9.5.3

8.5 Процедура соединения

8.5.1 Общие положения

В 8.5.2—8.5.5 описываются условия входа, нормальные процедуры, условия выхода, и условия ошибок, которые могут возникнуть в состоянии Соединен в диаграммах состояний.

8.5.2 Условия входа

Агент и менеджер переходят в состояние Соединен, когда транспортный уровень указывает на то, что соединение между агентом и менеджером установлено. И агент и менеджер получают индикацию соединения из собственных транспортных уровней (т. е. ни одной коммуникации на прикладном уровне в это время не происходит). В начальный момент перехода в состояние Соединен агент и менеджер запускаются из состояния Неассоциированный, подсостояния состояния Соединен.

8.5.3 Нормальные процедуры

Так как состояние Соединен имеет ряд подсостояний, фактические рабочие условия описаны как часть этих подсостояний.

8.5.4 Условия выхода

Агент и менеджер должен выйти из ассоциированного состояния, перейдя в состояние разъединения, отправив запрос на завершение ассоциации, и ожидая ответ на завершение ассоциации. Затем агент и менеджер должны закрыть активную ассоциацию и вернуться в неассоциированное состояние. Это обычные действия, перед тем, как агент или менеджер выйдет из состояния Соединен. Затем за закрытие соединения отвечает транспортный уровень.

8.5.5 Условия возникновения ошибок

Передача данных может отключиться неожиданно (например, беспроводная передача данных может быть перемещена из диапазона, или кабельное сопряжение может быть преждевременно удалено). В этих случаях, передача должна предупредить транспортный уровень о разъединении. Агент и менеджер должны нести ответственность за возвращение в состояние разъединения. Это требование распространяется на состояние соединения и все подсостояния.

8.6 Неассоциированная процедура

8.6.1 Общие положения

В 8.6.2—8.6.5 описываются условия входа, нормальные процедуры, условия выхода, и условия ошибок, которые могут возникнуть для состояния не ассоциирован в диаграммах состояния.

8.6.2 Условия входа

Состояние не ассоциирован — это состояние по умолчанию, в которое входят всегда, когда агент или менеджер впервые уведомляет об установлении соединения. В это состояние также возвращаются каждый раз, когда агент или менеджер завершают или разрывают ассоциацию с партнером.

8.6.3 Нормальные процедуры

Обычно агент не делает ничего в данном состоянии. Менеджер ждет в данном состоянии, пока не получит сообщение с запросом на ассоциацию.

8.6.4 Условия выхода

Каждый раз, когда агент определяет, что он желает попытаться установить ассоциацию с менеджером, он переходит в состояние Ассоциирования. Менеджер переходит в данное состояние, после получения сообщения с запросом на ассоциацию.

8.6.5 Условия возникновения ошибок

При неассоциированном состоянии может возникнуть ряд ошибок. Реакцией на возникновение таких условий может быть либо игнорирование условия либо создание сообщения о разрыве ассоциации. См. таблицу Е.1, состояние 2 (неассоциированное состояние) для получения дополнительной информации.

8.7 Ассоциированная процедура

8.7.1 Общие положения

Ассоциирующая процедура позволяет агенту и менеджеру согласовать протокол общих данных и общий набор рабочих параметров.

8.7.2 Условия входа

И агент и менеджер должны оставаться в состоянии Не ассоциирован до тех пор, пока агент не определит, что ассоциация является желательной. На этом этапе агент должен перейти в состояние Ассоциации и отправить запрос на ассоциацию. Менеджер переходит в состояние Ассоциации после получения запроса на ассоциацию от агента.

8.7.3 Нормальные процедуры

Рисунки 12 и 13 изображают диаграмму последовательности для ассоциирующей процедуры между агентом и менеджером. На рисунке 12 показана ситуация, при которой менеджер уже знает конфигурацию агента в связи с предварительным соединением с ним, или в связи с тем, что агент имеет стандартную конфигурацию (т. е. стандартную конфигурацию, указанную в стандарте специализации). На рисунке 13 показана ситуация, при которой менеджер не знает конфигурацию агента и информирует его о том, что запрос на ассоциацию будет принят, но конфигурация неизвестна.

В 8.7.3.1—8.7.3.2 описываются условия работы для ролей двух разных устройств: агента и менеджера.

8.7.3.1 Процедура агента

8.7.3.1.1 Общие положения

Когда агент намерен создать ассоциацию, он должен начать с перехода в состояние Ассоциации и отправить сообщение с запросом ассоциации менеджеру. Определение AarqApdu (см. А.8) поясняет формат сообщения с запросом ассоциации. Пример запроса ассоциации см. в Н.2.1.1.

Сообщение с запросом ассоциации содержит следующие пункты:

- версию используемого протокола ассоциации (assoc-version). Это поле позволяет агенту и менеджеру убедиться в том, что они используют одну и ту же версию протокола обмена;

- перечень протоколов передачи данных, которые поддерживает агент (data-proto-list). Агенту разрешено поддерживать один или несколько протоколов передачи данных для обмена информацией. Агент должен заказать перечень протоколов передачи данных, начиная с наиболее предпочтительного протокола, и заканчивая наименее предпочтительным протоколом.

Менеджер выбирает предпочтительный протокол и сообщает об этом агенту.

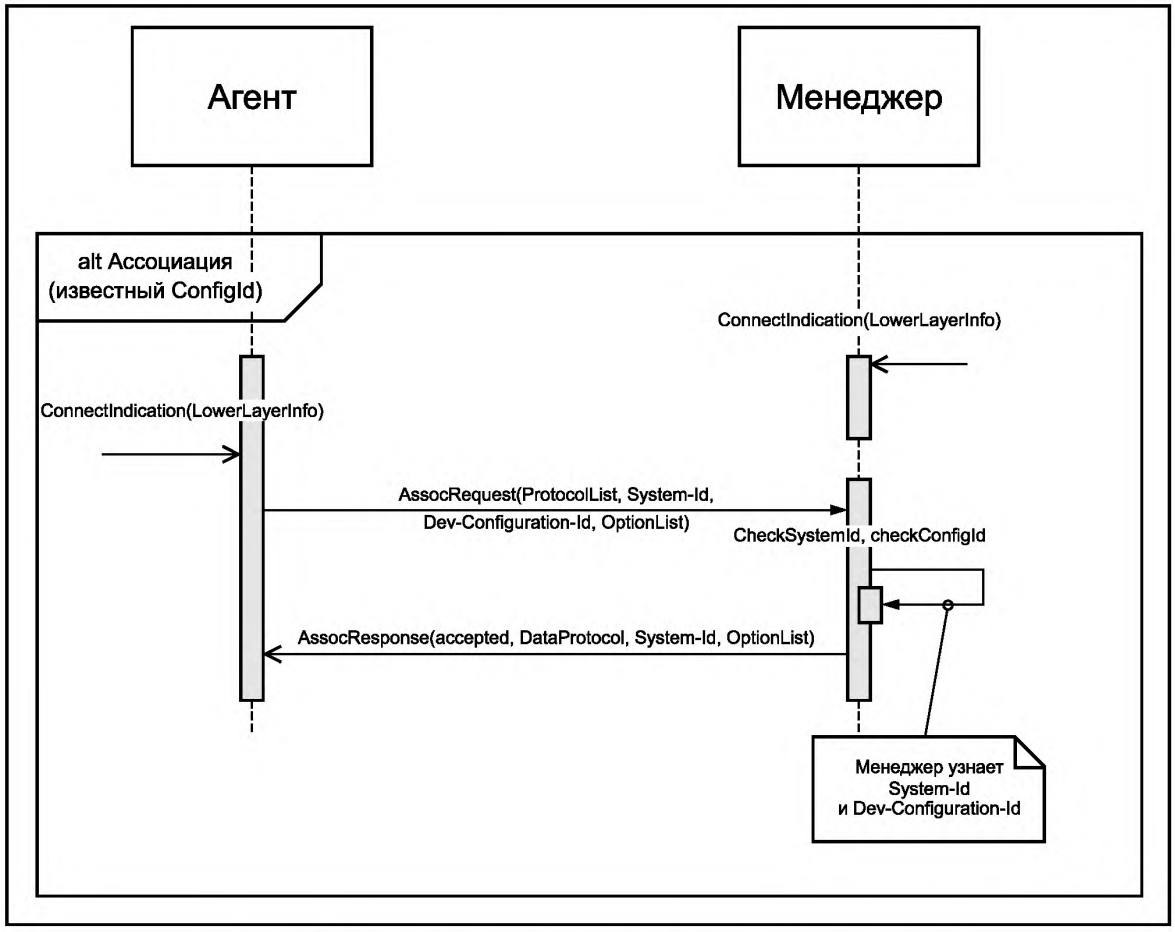


Рисунок 12 — Ассоциированная процедура (известная конфигурация)

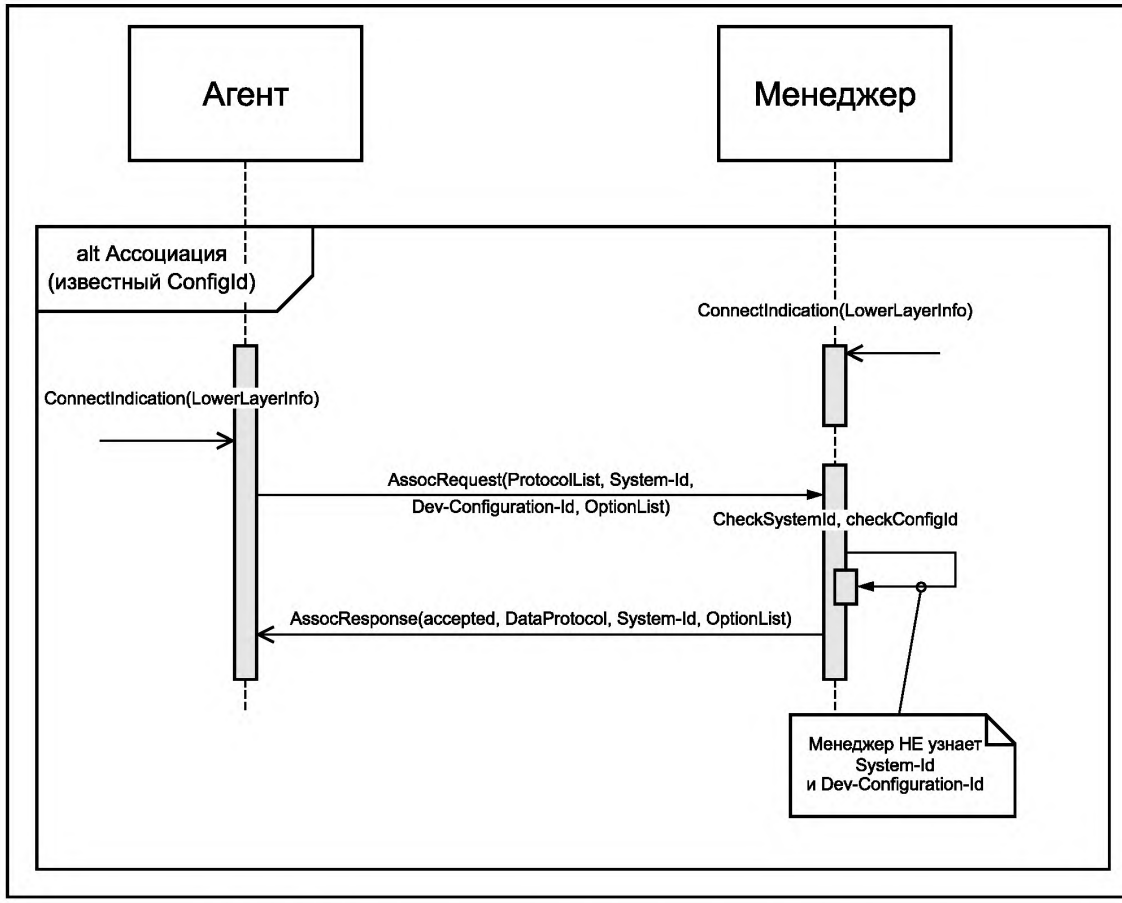


Рисунок 13 — Ассоциированная процедура (неизвестная конфигурация)

Чтобы разрешить выбор протокола передачи данных в ходе ассоциации, перечень data-proto-list содержит идентификатор, который обозначает, что протокол данных определяется одним стандартом из серии ИСО/ИИЭР 11073 или определяется производителем. Эти параметры описаны в двух следующих подразделах. Дополнительные коды доступны, но сохранены для будущих расширений.

8.7.3.1.2 Протокол обмена данными, определенный настоящим стандартом

Если агент устанавливает data-proto-id в A.8 до data-proto-id-20601, то он должен придерживаться абстрактных определений синтаксиса, содержащихся в настоящем стандарте для типов данных и обмена сообщениями. Кроме того, поле data-proto-info заполняется вместе со структурой PhdAssociationInformation, которая определяет следующую информацию:

- версия протокола обмена данными;
- конкретное правило(а) кодирования DataApdu, поддерживающиеся агентом. Агент должен установить один или несколько битов encoding-rules:
- агент должен всегда поддерживать правила MDER (правила кодирования медицинских приборов), т. е. бит MDER в поле encoding-rules должен устанавливаться агентом;
- агент может предложить другие правила кодирования, помимо MDER, менеджеру, установив другие биты в поле encoding-rules;
- версия использованной номенклатуры;
- поле, указывающее все функциональные блоки и дополнительные свойства, поддерживаемые агентом;
- тип системы (в данном случае агента);

- уникальный идентификатор System-Id (см. таблицу 2) агента. Формат EUI-64 используется для идентификации агента. Менеджер может использовать это поле для определения идентификации агента, с которым он осуществляет коммуникацию и, возможно, для реализации простой политики ограничения доступа;

- dev-config-id, обозначающий текущую конфигурацию агента, описан в разделе 7.4.3. Значение dev-config-id для стандартных конфигураций должно находиться между standard-config-start и standard-config-end, включительно. Для расширенных конфигураций, значение dev-config-id должно находиться между extended-config-start and extended-config-end включительно;

- data-req-mode-capab,который определяет режимы запроса данных, поддерживаемые агентом (см. 8.9.3.3.3);

- option-list, который содержит список дополнительных атрибутов агента, намеренного начать коммуникацию.

8.7.3.1.3 Протокол обмена данными, определенный производителем

Другие спецификации могут использовать первоначальный запрос на ассоциацию с целью договориться об использовании протоколов, определенных производителем. В этом случае агент устанавливает data-proto-id в A.8 до data-proto-id-external. Чтобы различать многие возможные протоколы, определенные изготовителем, агент использует информационную структуру ManufSpecAssociation, чтобы предоставить УУИд (универсально уникальный идентификатор), который обозначает конкретный протокол.

Реальное поведение протокола за пределами изначальной ассоциации находится вне сферы действия серии стандартов ИСО/ИИЭР 11073. УУИд должен быть сформирован в соответствии с ITU-T Запись. X.667 (сентябрь 2004).

8.7.3.2 Ответ на ассоциацию

После того как агент отправит сообщение с запросом ассоциации, он должен ждать получения сообщения с ответом на ассоциацию от менеджера или тайм-аута (см. 8.7.5 для получения информации по условиям тайм-аута).

Определение AareApdu (см. A.8) описывает формат сообщения с ответом на ассоциацию. Пример ответа на ассоциацию можно найти в H.2.1.2. Сообщение с ответом на ассоциацию содержит следующее:

- поле результата, представляющее исход процедуры объединения;

- версия общего протокола данных, выбранного менеджером, если поле результата эквивалентно accepted или accepted-unknown-config;

- одно и только одно правило кодирования DataApdu, выбранное менеджером, если поле результата эквивалентно accepted или accepted-unknown-config;

- менеджер должен всегда поддерживать правила MDER для обеспечения способности к взаимодействию;

- кроме того, менеджер может выбрать одно из других правил кодирования, помимо правил MDER, предложенных агентом.

Примечание — MDER всегда поддерживается как агентом, так и менеджером. Однако если агент предлагает дополнительные правила кодирования менеджеру, можно сделать вывод, что у агента была на то веская причина (т.е. разработка дополнительной поддержки правил кодирования не выполняется без веских причин продукта). Таким образом, если агент предлагает дополнительные правила кодирования помимо MDER, предполагается, что менеджер выполнит одно из дополнительных предложенных правил кодирования, если это возможно. Например, если агент предлагает MDER и правила уплотненного кодирования (PER), предполагается, что менеджер выполнит кодировку PER, если это возможно. Если агент предлагает MDER и правила кодирования XML (XER — Правила кодировки расширяемого языка разметки), предполагается, что менеджер выполнит правила кодирования XER, если это возможно. На случай, если агент предлагает MDER, PER и XER, этот стандарт не дает рекомендаций относительно выбора предпочтительного правила кодирования;

- версия номенклатуры, выбранная менеджером, если поле результата эквивалентно accepted или accepted-unknown-config;

- тип системы (в данном случае менеджера, так как сообщение посылается менеджером);

- уникальный идентификатор системы (см. таблицу 2) менеджера. EUI-64 используется для уникальной идентификации менеджера. Агент может использовать это поле для определения того, установлена ли связь с требуемым менеджером;

- поле dev-config-id должен выглядеть как manager-config-response в ответе;

- Data-req-mode-capab должен быть пустым в ответе;

- поле, указывающее общие функциональные единицы и дополнительные функции, которые выбираются менеджером, если поле результатов эквивалентно accepted или accepted-unknown-config.

Поле результата в сообщении с ответом на ассоциацию указывает результат запроса. Возможные исходы (см. AssociateResult в А.8) могут быть следующими:

- accepted означает, что ассоциация принимается и конфигурация известна. Агент должен перейти в рабочее состояние (см. 8.9 для получения подробной информации об эксплуатационных процедурах);

- accepted-unknown-config означает, что конфигурация принята, но агенту необходимо направить свою конфигурацию менеджеру. Когда агент получает ответ с сообщением о том, что конфигурация неизвестна, он должен перейти в состояние конфигурации и следовать процедурам, описанным в разделе 8.7.6 для передачи конфигурации;

- rejected-unsupported-assoc-version означает, что агент и менеджер не разделяют общую версию ассоциации;

- rejected-no-common-protocol означает, что менеджер отклоняет запрос на ассоциацию, потому что не найден единый протокол данных в перечне DataProtoList, разделенном между менеджером и агентом;

- rejected-no-common-parameter означает, что менеджер отклоняет запрос на ассоциацию, потому что менеджер и агент не имеют общего набора рабочих параметров в информации об ассоциации, специфичной для протокола (PhdAssociationInformation);

- rejected-unauthorized используется, когда менеджер определяет, что агент не авторизован для подключения. Способ принятия решения устанавливается поставщиком;

- rejected-transient используется, когда менеджер не может принять ассоциацию из-за переходных режимов, таких как ограниченность ресурсов;

- rejected-permanent означает, что менеджер не может общаться с агентом, но никакой дальнейшей информации касательно причины не доступно;

- rejected-unknown следует использовать с осторожностью и только тогда, когда вышеуказанные коды возврата не применяются.

При всех rejected-* условиях, агент должен перейти в неассоциированное состояние.

8.7.3.3 Процедура менеджера

Когда менеджер получает запрос ассоциации, он должен сравнить параметры протокола и эксплуатационные параметры с собственными параметрами и определить, является ли агент совместимым с менеджером. Если соединение является двунаправленным, менеджер должен доложить о результатах этой оценки в поле результатов в ответе на ассоциацию.

Возможные причины отклонения перечислены в пункте 8.7.3.2. Если менеджер отклоняет ассоциацию, он должен перейти в неассоциированное состояние.

Если запрос не отклоняется менеджером, поле результата в сообщении с ответом на ассоциацию, исходящее от менеджера, указывает на то, понимает ли менеджер конфигурацию. Если менеджер признает dev-config-id в качестве известного стандартной специализации устройства или как предыдущую ассоциацию, менеджер высылает сообщение с ответом на ассоциацию с указанием «accepted» в поле результата и переходит в рабочее состояние.

Если менеджер не признает dev-config-id, менеджер высылает сообщение с ответом на ассоциацию с указанием accepted-unknown-config в поле результата и переходит в конфигурирующее состояние.

Когда менеджер принимает общий протокол, то в ответе на ассоциацию он должен вернуть предпочтительный общий протокол передачи данных и общий набор рабочих параметров, выбранных из списка, предоставленного в запросе на ассоциацию.

8.7.4 Условия выхода

Менеджер выходит после отправки ответа на ассоциацию. Агент выходит из ассоциирующего состояния после получения ответа на ассоциацию.

8.7.5 Условия возникновения ошибок

Агент должен ждать сообщения с ответом на ассоциацию в течение периода TOassoc (тайм-аут: процедура ассоциация). Если период TO_{assoc} истекает, агент должен повторно передать сообщение с запросом ассоциации до RC_{assoc} раз (счётчик повторов: процедура ассоциации) после первого тайм-аута, с периодом TO_{assoc} между каждым последующим сообщением. Если после данной последовательности повторных попыток агент не может успешно получить любые сообщения с ответом на ассоциацию, то он направляет сообщение о разрыве ассоциации с менеджером и переходит обратно в неассоциированное состояние.

Если агент или менеджер получает сообщение о разрыве ассоциации, находясь в ассоциированном состоянии, он должен перейти в неассоциированное состояние.

8.7.6 Тестовая ассоциация

Тестовой ассоциацией является ассоциация, созданная агентом и менеджером с целью обмена данными, которые предназначены для испытаний. Этот стандарт не определяет, что эти обмены не схожи с семантикой, ассоциированной с ними, а только с процессом, при котором устройства входят и выходят в тестовую ассоциацию. Отдельные специализации устройств могут определить стандартизированные ресурсы испытаний, идентификаторы конфигурации и процессы, которые могут быть использованы во время тестовой ассоциации. Тестовая ассоциация может быть использована в целях испытания, специфичных для производителя.

Поскольку этот стандарт не определяет семантику тестовой ассоциации, он также не определяет конкретные механизмы для того, чтобы тестовые данные управлялись должным образом. Тем не менее, очень важно, чтобы устройства обеспечивали защиту с целью убедиться в том, что данные испытаний не обрабатываются другими организациями как фактические данные измерений. В общем, только те элементы, которые понимают концепцию тестовой ассоциации, должны видеть данные измерений, созданные тестовой ассоциацией. Разработчикам следует предпринять следующие шаги:

- установить бит test-data или бит demo-data атрибута MeasurementStatus при создании моделированных данных измерений. Если атрибут MeasurementStatus не поддерживается, должны быть использованы альтернативные средства выделения таких данных;

- убедиться в том, что местные устройства индикации и хранилища данных измерений игнорируют данные испытаний (test-data) и демо-данные (demo-data), если они не могут должным образом выделить такие данные для пользователя и не могут обнаружить вход и выход из тестовой ассоциации. Местный компонент на агенте, не участвующий в протоколе ИИЭР 11073-20601 не может являться подходящим вариантом для получения данных измерения испытания;

- убедиться в том, что данные измерений, помещенные в хранилище РМ или другую постоянную структуру хранилища, никогда не выводятся за пределы тестовой ассоциации. Для данной цели может быть использована маркировка и/или очистка постоянной памяти;

- убедиться в том, что устройства, которые отображают или сохраняют данные испытания или демо-данные должным образом обновлены, когда такие события, как отключение, вызывают необходимость завершения тестовой ассоциации;

- для того чтобы тестовая ассоциация была сформирована, менеджер и агент должны поддерживать тестовые ассоциации, и оба должны быть готовы вступить в тестовую ассоциацию в заданный момент времени. Трехэтапный протокол используется для однозначного вступления в тестовую ассоциацию.

На первом этапе агент передает менеджеру два бита информации в поле fun-units структуры PhdAssociationInformation. Бит fun-unit-createtestassociation указывает, что агент имеет возможности для испытаний, которые можно использовать в тестовой ассоциации. Бит fun-unit-createtestassociation используется агентом в качестве запроса менеджеру на установку тестовой ассоциации. Агент не должен устанавливать бит fun-unit-createtestassociation, если он также не установил бит fun-unit-havetestcap. Если агент вводит бит fun-unit-havetestcap в структуру PhdAssociationInformation, он не должен завершать ассоциацию вследствие получения ответа с установленным битом fun-unit-createtestassociation. Это означает, что если агент устанавливает бит fun-unit-havetestcap и предлагает более одной конфигурации, в которых определены стандартизированные возможности испытания, агент должен быть готов вступить в тестовую ассоциацию, используя одну из этих конфигураций.

На втором этапе протокола менеджер обратно сигнализирует агенту о своем намерении установить тестовую ассоциацию. Менеджер передает эту информацию агенту через бит fun-unit-createtestassociation. Бит устанавливается менеджером для того, чтобы указать, что он вступил в тестовую ассоциацию. Менеджер должен установить этот бит, только в случае если fun-unit-havetestcap установлен в запросе от агента. В соответствии с данным стандартом менеджер не обязан входить в тестовую ассоциацию даже по просьбе агента. Агент должен игнорировать бит fun-unit-havetestcap в ответе на ассоциацию.

Заключительный шаг протокола тестовой ассоциации включает в себя решение агента о продолжении тестовой ассоциации или ее завершении. Агент не вступает в состояние тестовой ассоциации, если менеджер не установил бит fun-unit-createtestassociation. Тестовая ассоциация завершается, когда машина состояний ассоциации входит в неассоциированное состояние.

8.8 Процедура конфигурирования

8.8.1 Общие положения

Конфигурирующее состояние возникает, когда агенту необходимо передать информацию о конфигурации менеджеру.

8.8.2 Условия входа

Сообщение с ответом на ассоциацию с результатом accepted-unknown-config должны инициировать агента войти в конфигурирующее состояние и отправить его конфигурацию менеджеру. Менеджер входит в конфигурирующее состояние сразу же после отправки ответа на ассоциацию с результатом accepted-unknown-config.

Обратите внимание на то, что часть конфигурации также является присваиванием значения атрибута дескриптора экземпляру объекта. Если менеджер узнает конфигурацию агента, он также узнает присвоенные значения атрибута дескриптора. Это означает, что стандартная конфигурация, например, конфигурация, определенная в специализации устройства ИСО/ИИЭР 11073-104zz, определяет фиксированные значения для атрибутов дескриптора.

8.8.3 Нормальные процедуры

На рисунке 14 показана схема последовательности для процедуры конфигурации. Во время процедуры конфигурации агент передает информацию о конфигурации всех объектов, которые он поддерживает, за исключением объекта MDS, а также все статические атрибуты в объектах. Агенты, как правило, имеют высокий уровень статичности конфигурации, таким образом, передача всех статических частей во время разовой фазы конфигурирования уменьшает общий коммуникационный трафик. Новые типы измерений не добавляются динамически, многие атрибуты не изменяются, а наборы переданных атрибутов объекта часто остаются прежними. Реконфигурация требуется только при изменении агента (например, как часть начальной процедуры установки, где специфические возможности измерения могут быть конфигурированы).

Агент выполняет процедуру конфигурации с помощью сообщения Подтвержденный Запрос События с событием MDC_NOTI_CONFIG для того, чтобы отправить свою конфигурацию менеджеру. Сообщение — уведомление о конфигурации указывает на:

- все объекты, поддерживаемые агентом, за исключением объекта системы MDS, и
- набор статических атрибутов для каждого объекта.

Атрибуты включают идентификацию номенклатуры классов объекта (см. 6.3.4.2, 6.3.5.2 и 6.3.6.2), физиологический идентификатор (код номенклатуры), идентификатор единицы/размера (код номенклатуры), дополнительно, строки для маркировки, а также любые другие статические атрибуты, которые могут быть полезными. Эта информация рассматривает плоское (неиерархическое) и статическое дерево состава агента. Объект системы MDS исключается из конфигурации, так как большая часть информации является динамической или специфичной для производителя. Отдельная команда Get MDS Object (Получение объекта системы MDS) побуждает механизм загружать данную информацию (см. 6.3.2.6.1).

Для объектов, о которых сообщается на тех же атрибутах каждый раз, рекомендуется отчет о событии фиксированного формата (см. 7.4.5), и агент должен отправить карту Attribute-Value-Map, описывающую структуру сообщения. Касательно объектов сканера, которые используют отчеты о событиях группированного формата, агент направляет карту andle-Attr-Val-Map с описанием структуры.

Если набор переданных атрибутов объекта не является фиксированным, рекомендуется отчет о событии переменного формата. Этот формат позволяет сообщать атрибуты конфигурации как часть обновления значений. В этом случае Attribute-Value-Map не предоставляется в отчете о событии конфигурации или предоставляется как пустой список.

Агент должен использовать сообщение данных «Remote Operation Invoke | Confirmed Event Report» (см. A.10.3 для начального определения EventReportArgumentSimple) с типом события MDC_NOTI_CONFIG при передаче своей конфигурации (см. ConfigReport в A.11.5 для остальной части структуры). Менеджер должен ответить сообщением «Remote Operation Response | Confirmed Event Report» (см. A.10.3 для определения EventReportResultSimple) с указанием типа события MDC_NOTI_CONFIG, внесенного в структуру ConfigReportRsp. См. H.2.2 для получения примера запроса события конфигурации, отправленного агентом, за которым следует пример ответа от менеджера.

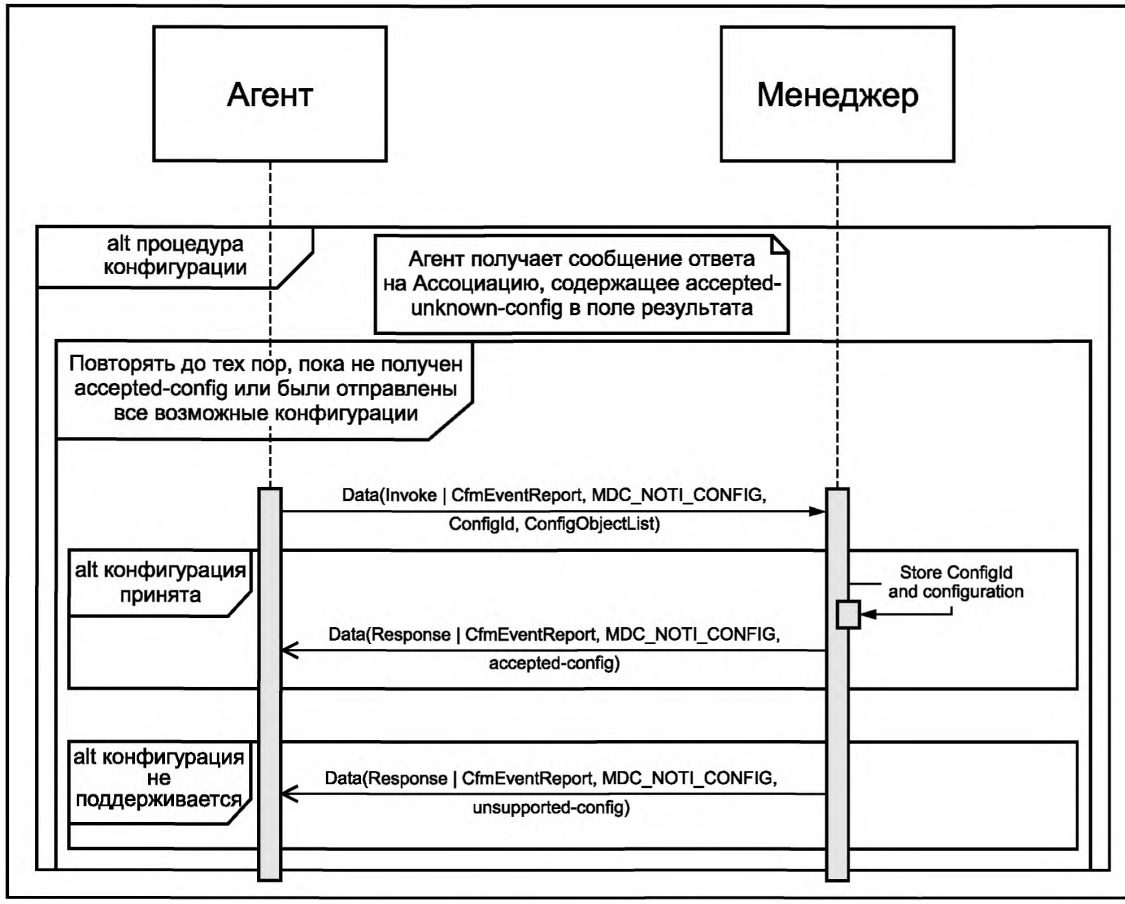


Рисунок 14 — Процедура конфигурации

Агенты могут поддерживать более одной конфигурации. В этом случае агент должен послать каждую из имеющихся конфигураций, начиная с предпочтительной конфигурации. Если менеджер принимает конфигурацию, он отвечает сообщением accepted-config, затем менеджер и агент переходят в рабочее состояние. Если менеджер не принимает конфигурацию, то он должен отправить ответ unsupported-config. По получении ответа unsupported-config агент отправляет следующую конфигурацию. Этот процесс повторяется до тех пор, пока агент не попытается отправить все конфигурации. Затем он должен отправить сообщение о выпуске ассоциации с кодом причины no-more-configurations,с целью указать, что он не может работать с менеджером.

Агент, который соответствует одному или нескольким специализациям устройств, которые определяют стандартные конфигурации (то есть специализации ИСО/ИИЭР 11073-104zz) должен поддерживать одну или несколько стандартных конфигураций и может поддерживать одну или несколько расширенных конфигураций. Для совместимости этот агент направляет поддерживаемые стандартные конфигурации как запасные, если расширенные конфигурации не поддерживаются.

Если агент соответствует стандартной конфигурации, то он должен использовать dev-config-id, как определено в конкретной специализации устройства ИСО/ИИЭР 11073-104zz. Эти значения dev-config-id стандартной конфигурации назначаются в диапазоне между standard-config-start и standard-config-end включительно. Когда агент предоставляет dev-config-id, соответствующий стандартной конфигурации, сообщение о конфигурации не должно содержать информацию о конфигурации и тип события MDC_NOTI_CONFIG может быть отправлен со стандартным идентификатором конфигурации и пустым списком ConfigObjectList. Если менеджер не узнает стандартную конфигурацию (например, менеджер был выпущен перед тем, как была выпущена специализации устройства), то он должен от-

править ответ standard-config-unknown. Агент может повторить конфигурацию для стандартного устройства, отправив полную информацию о конфигурации.

Агент, имеющий нестандартную конфигурацию, должен присвоить уникальный идентификатор своей конфигурации путем создания значения для dev-config-id в диапазоне между extended-config-start и extended-config-end, включительно.

Агент может использовать то же значение для dev-config-id в последующих запросах на ассоциацию с менеджером, чтобы обозначить ту же конфигурацию устройства. Выбранное значение dev-config-id должно быть сообщено в атрибуте Dev-Configuration-Id объекта MDS.

Если агент изменяет свою конфигурацию так, что больше не может поддерживать старую, или определяет, что новая конфигурация должна использоваться как предпочтительная, он должен закрыть любую существующую ассоциацию, отправив сообщение о выпуске ассоциации с указанием причины изменения конфигурации (configuration-changed). Если новая конфигурация является новой расширенной конфигурацией, агент должен назначить новый идентификатор конфигурации. В следующий раз, когда агент ассоциируется, он совещается с менеджером путем пошаговой отладки конфигурации в порядке приоритетности, как описано выше.

8.8.4 Условия выхода

Когда менеджер принимает предпочтительную конфигурацию, он должен отправить ответ accepted-config агенту и перейти в рабочее состояние. Если менеджер получает запрос на выпуск ассоциации с указанием причины no-more-configurations, чтобы указать, что агент не имеет следующих конфигураций, менеджер должен перейти в неассоциированное состояние.

Когда агент получает ответ accepted-config от менеджера, он должен перейти в рабочее состояние. Если агент получает ответ от менеджера unsupported-config, он должен посылать следующую конфигурацию менеджеру до тех пор, пока не останется больше конфигураций. Тогда он должен послать сообщение с запросом на выпуск ассоциации с указанием причины no-more-configurations и войти в неассоциированное состояние.

8.8.5 Условия возникновения ошибок

Агент должен ожидать получения сообщения «Remote Operation Invoke | Confirmed Event Report» MDC_NOTI_CONFIG в течение периода TO_{config} (время выполнения процедуры конфигурации). Если период TO_{config} истекает, агент направляет сообщение о разрыве ассоциации менеджеру и переходит обратно в неассоциированное состояние.

Менеджер должен ждать минимум TO_{config} секунд в состоянии ожидания конфигурации для получения информации о конфигурации, перед отправкой сообщения о разрыве ассоциации и переходом обратно в неассоциированное состояние.

Если агент или менеджер получает или посылает сообщение о разрыве ассоциации в любое время, он должен перейти в неассоциированное состояние.

8.9 Процедура Выполнения

8.9.1 Общие положения

Передача данных о текущем состоянии и информации о статусе агента происходит во время процедуры Выполнения.

8.9.2 Условия входа

Агент и менеджер входят в рабочее состояние, когда конфигурация агента уже известна менеджеру или после того, как агент передал приемлемую конфигурацию менеджеру.

8.9.3 Нормальные процедуры

8.9.3.1 Общие положения

В 8.9.3.2—8.9.3.4 описываются процедуры, которые могут возникнуть при реализации процедуры Выполнения.

8.9.3.2 Атрибуты объекта системы MDS

В любое время в рабочем и ассоциированном состоянии менеджер может запросить атрибуты объекта системы MDS агента, отправив сообщение с данными, по команде «Remote Operation Invoke | Get» и полученное значение дескриптора 0. Агент должен сообщить менеджеру о введении им атрибута объекта системы MDS, используя сообщение данных с ответом «Remote Operation Response | Get». См. H.2.3 для получения примера использования данного набора сообщений. Агенты должны поддерживать команду Get, которая запрашивает все атрибуты (т. е. список attribute-id-list пуст). Агенты могут поддерживать извлечение конкретного перечня идентификаторов атрибута.

На рисунке 15 показана схема последовательности запроса атрибутов объекта системы MDS менеджером от агента.

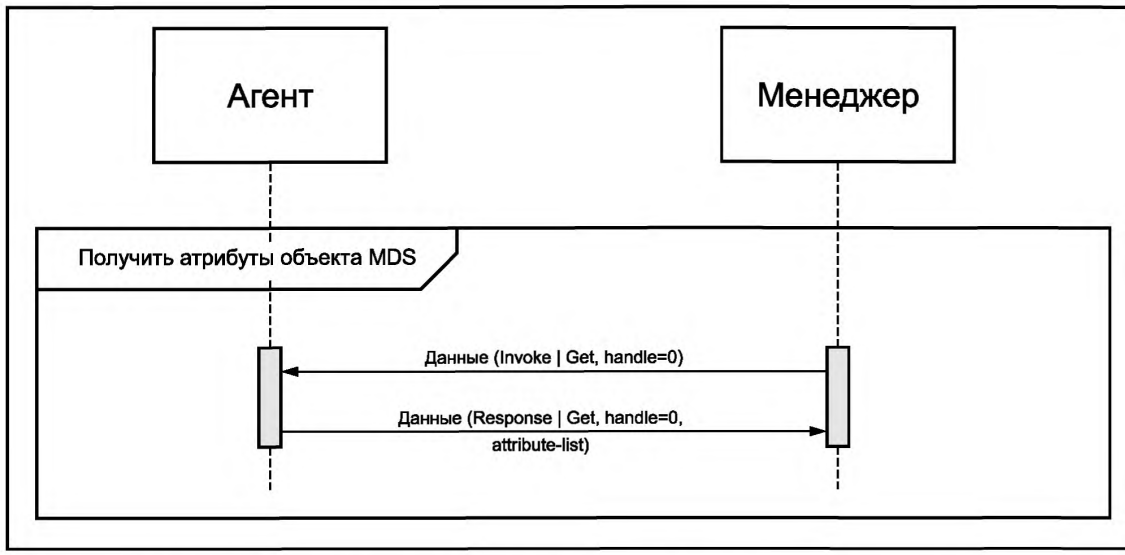


Рисунок 15 — Схема последовательности получения атрибутов объекта системы MDS

8.9.3.3 Передача данных измерений

8.9.3.3.1 Общие положения

Передача данных измерений может быть инициирована как агентом, так и менеджером, как указано в пункте 7.4.4. Передача, инициированная агентом, как правило, ожидается от агентов, которые передают небольшое количество редкой эпизодической информации или требуют минимальной пропускной способности. Агентам с большими объемами данных, частыми передачами данных или потоковыми данными следует использовать передачу, инициированную менеджером. Передача, инициированная менеджером, является более предпочтительной во всех случаях, так как этот подход предоставляет механизм контроля потока данных. Обратите внимание на то, что получение запроса на передачу данных измерений не является командой к тому, чтобы агент выполнил измерение, а к тому, что он должен передать любые доступные данные измерений.

В каждом случае, за исключением отдельного режима ответа, передача данных измерений осуществляется с использованием отчета о событиях, подтвержденного или неподтвержденного по выбору агента.

Все варианты двух методов, подробно описаны в пунктах 8.9.3.3.2 — 8.9.3.3.8.

8.9.3.3.2 Передача данных измерений, инициированная агентом.

Когда агент поддерживает передачу, инициированную агентом, он должен указать, что поддерживает через структуру DataReqModeCapab или иметь один или несколько экземпляров объекта сканера в конфигурации агента.

Агент

Объекты сканера должны начинаться с атрибута рабочего состояния, установленного для блокировки на агентах с двунаправленной связью, пока менеджер не разблокирует их. Менеджер должен установить состояние объектов сканера для их разблокировки, когда он хочет получить данные.

Для передачи данных измерений, инициированной агентом, поле data-req-id в отчете сканера (MDC_NOTI_SCAN_REPORT_ *) должно быть установлено на data-req-id-agent-initiated.

Менеджер может остановить передачу данных измерений, инициированную агентом от агента, отправив запрос на выпуск ассоциации или сообщение о разрыве ассоциации агенту для того, чтобы прекратить связь. Если агент использует объекты сканера, менеджер может заблокировать сканер с помощью сервиса SET на атрибуте состояния Выполнения.

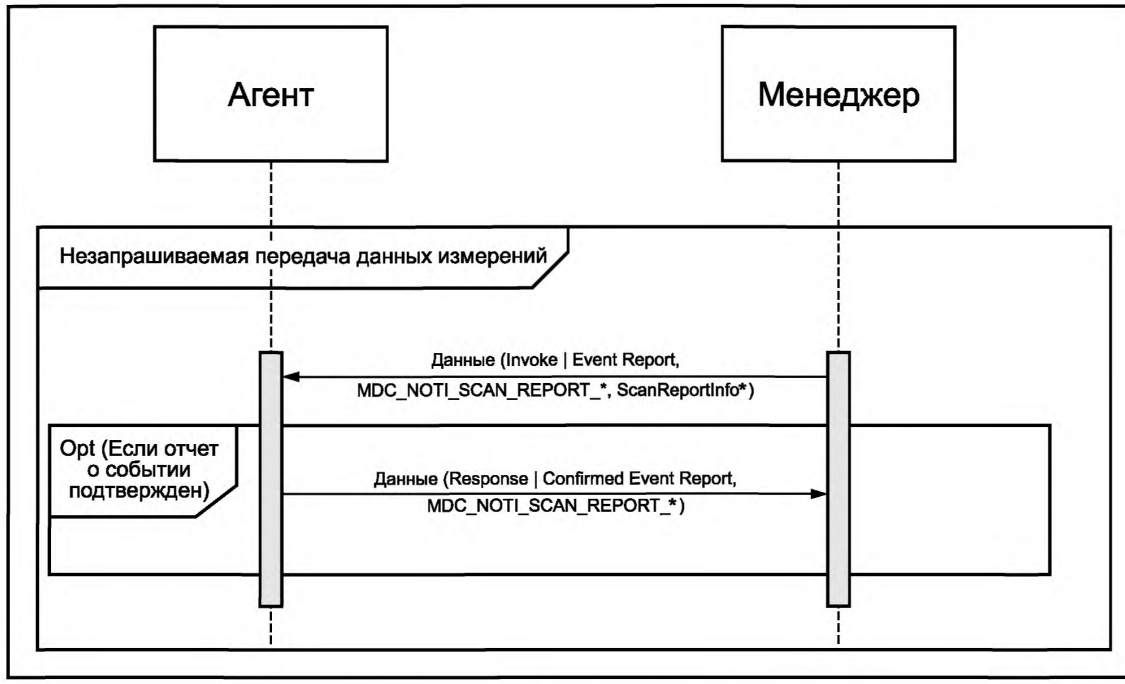


Рисунок 16 — Передача данных измерений, инициированная агентом

8.9.3.3.3 Обзор передачи данных измерений, инициированной менеджером

Когда агент поддерживает передачу, инициированную менеджером, он должен указать, какие функции он поддерживает, используя структуру DataReqModeCapab. Если агент не предоставляет DataReqModeCapab, менеджер должен допускать, что ни одно из свойств не поддерживается агентом.

При передаче данных измерений, инициированной менеджером, менеджер использует сервис ACTION (см. 7.3), предоставленный агентом, с целью запросить передачу данных измерений у агента. Когда менеджер намерен сделать это, он должен послать подтвержденный запрос DataApdu ActionArgumentSimple с типом действия MDC_ACT_DATA_REQUEST, за которым следует информация DataRequest. Этот запрос данных может являться запросом на начало или запросом на остановку, в соответствии с указанием бита data-req-start-stop режима data-req-mode (см. A.11.5) или запросом на продолжение в соответствии с указанием бита data-req-continuation.

Для запроса на начало могут использоваться три режима: одиночный ответ (data-req-mode-single-rsp), период времени (data-req-mode-time-period), и без ограничения по времени (data-req-mode-time-no-limit). В зависимости от режима запроса на начало агент может отправить один или несколько отчетов о событиях менеджеру. Когда менеджер запускает режим передачи данных, он предоставляет идентификатор data-req-id, который должен использоваться агентом во всех отчетах событий. Если, в то время как режим передачи данных запущен, приходит новый запрос на начало с тем же data-req-id, этот запрос будет считаться приоритетным, а новый инициирован. Агент рассматривает новый запрос на начало, как если бы это была остановка с последующим началом. Режимы одиночного ответа и периода времени имеют четко определенные конечные точки, после чего ресурсы, поддерживающие эти запросы, могут быть выпущены. Запрос без ограничения по времени не имеет четко определенной конечной точки. Менеджер должен выпустить запрос на остановку, если больше не заинтересован в потоке измерения, в частности для запроса без ограничения по времени, для того чтобы высвободить ресурсы на агенте.

Для каждого из этих режимов может выбираться один из трех различных вариантов для сферы действия объекта, к которому относится запрос на передачу данных: все данные, имеющиеся у агента

(data-req-scope-all), данные, имеющиеся у агента в соответствии с конкретным классом объекта (data-req-scope-class), и данные, имеющиеся у агента в соответствии с конкретными объектами, определенными их дескрипторами (data-req-scope-handle).

При использовании data-req-scope-all агент должен рассмотреть все объекты, за исключением объекта системы MDS, при определении содержания каждого отчета о событии.

При использовании data-req-scope-class менеджер должен использовать data-req-class с целью определения класса объектов для создания отчета. При формировании отчетов о событиях агент должен рассматривать только те объекты, что описаны данным классом.

Идентификаторы разрешенного класса включают MDC_MOC_VMO_METRIC_NU, MDC_MOC_ VMO_ METRIC_SA_RT и MDC_ MOC_VMO_METRIC_ENUM.

Когда data-req-scope-handle отправлен, менеджер должен предоставить список дескрипторов в data-req-obj-handle-list. Агент должен рассматривать только объекты, описанные действительными дескрипторами в списке дескрипторов при создании отчета о событиях. В данном контексте термин «действительный» относится ко всем дескрипторам, ассоциированным с метрически-производными объектами (например, числовые, PT-SA (матрица отсчета часов реального времени) или перечисления) поддерживаемыми агентом.

Запрос на остановку может быть использован менеджером с целью завершить передачу данных измерения периода времени или без ограничения по времени, начатую ранее.

При использовании установленного по времени режима, если менеджер хочет продлить время, которое предоставлено агенту для передачи данных, менеджер должен установить бит data-req-continuation в режиме и установить data-req-time, до значения количества времени, отведенного агенту для непрерывной передачи.

Поле data-req-id в запросе данных используется для дифференциации ответов из нескольких запросов данных того же агента (если агент позволяет несколько одновременных запросов данных). Менеджер должен установить значение поля data-req-id на значение в диапазоне от data-req-id-manager-initiated-min до data-req-id-managerinitiated-max, включительно. Агент должен использовать то же значение data-req-id во всех ассоциированных отчетах о событиях.

Следует отметить, что менеджер может установить значение поля данных data-req-id на любое значение в пределах допустимого диапазона. Тогда агент не будет опираться на поле data-req-id для того, чтобы вывести, например, порядок, в котором различные запросы данных были созданы менеджером.

Потоковые агенты должны использовать передачу данных измерений, инициированную менеджером (или объекты сканера) для того, чтобы позволить менеджеру контролировать то, как он принимает данные. Менеджеры должны разблокировать потоковых агентов, как можно раньше, чтобы информация агента стала легко доступной.

Три режима передачи данных измерений менеджера, инициированной менеджером, описаны в пунктах 8.9.3.3.4—8.9.3.3.6.

8.9.3.3.4 Режим одиночного ответа, инициированный менеджером

Режим одиночного ответа позволяет менеджеру запрашивать данные у агента и получать их в ответном сообщении (см. рисунок 17). Не существует требований к тому, чтобы агент собирал какие-либо данные (например, надувка манжетки для измерения кровяного давления) для составления ответа. Если агент не имеет доступных данных, он возвращает пустой список данных. Если агент имеет данные и статус результата — data-req-result-no-error, он должен послать сообщение DataResponse, которое содержит статус результата запроса (DataReqResult), а также данные измерений (ScanReportInfo*). Это ответное сообщение должно закрыть доступ к данным измерений.

Режим одиночного ответа не позволяет агенту подтверждать то, что менеджер получает данные измерений. Там, где необходимо такое подтверждение, используется срочная команда со значением тайм-аута от 0 (см. 8.9.3.3.5).

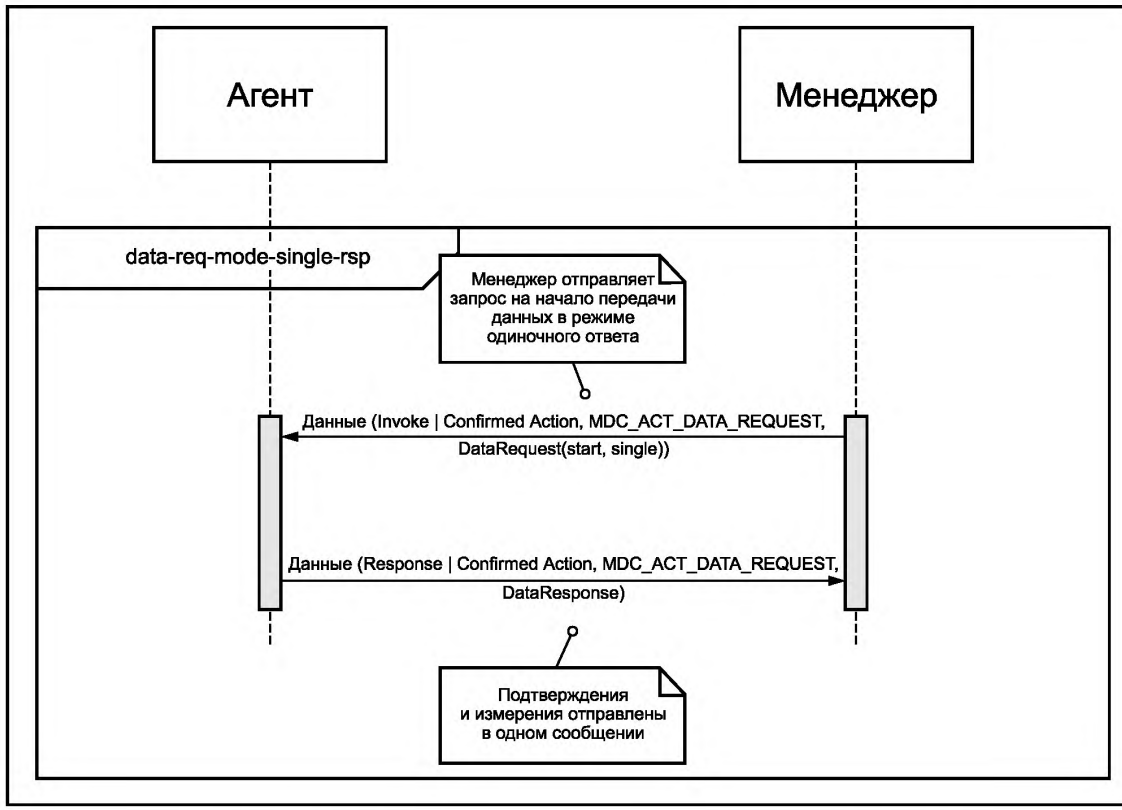


Рисунок 17 — Передача данных измерений, инициированная менеджером (data-req-mode-single-rsp)

8.9.3.3.5 Режим периода времени, инициированный менеджером

Режим периода времени используется менеджером для того, чтобы разблокировать агента для передачи данных, которые он собирает на протяжении указанного периода времени (см. рисунок 18). Когда агент получает начальное сообщение DataRequest от менеджера, агент должен отправить сообщение DataResponse, подтверждающее статус результата запроса (DataReqResult) без передачи каких-либо данных измерений в ответном сообщении. Если DataReqResult является data-req-result-no-error, каждый раз, когда данные становятся доступными, агент должен использовать сервис EVENT REPORT (Отчет о событиях) для оправки отчета(ов) о событии, содержащий данные измерений менеджеру, прежде чем истечет период времени, указанный в запросе данных, он получает запрос от менеджера на остановку или связь между агентом и менеджером прекращается. Агент определяет, использовать ли сообщение с подтвержденным или неподтвержденным отчетом о событии для передачи данных.

Если менеджер хочет продлить период времени, он должен перейти в data-req-id, установить data-req-continuation в режиме, и установить data-req-time, до такого значения времени, чтобы агент мог продолжить передачу данных. Все остальные параметры в DataRequest должны игнорироваться, и должны использоваться настройки исходной команды начала. Агент должен применять каждый новый период времени, измеренный с момента принятия команды. Если команда на продолжение получена для data-req-id, который не функционирует в заданном режиме, агент должен ответить data-req-result-continuation-not-supported. Если команда на продолжение получена для несуществующего data-req-id, агент должен ответить data-req-result-invalid-req-id. Например, если время истекает до приема команды продолжения, data-req-id останавливается и удаляется.

В режиме установленного периода времени, если data-req-time установлен на 0, агент должен подтвердить запрос. В случае подтверждения немедленно начните передачу любых данных, доступных

в настоящее время в отчетах событий, а затем остановите. В отличие от режима одиночного ответа (см. 8.9.3.3.4),режим установленного периода времени позволяет агенту использовать сообщения с подтвержденными или неподтвержденными отчетами о событиях. Например, агент может использовать подтвержденный отчет о событии, чтобы убедиться, что данные были получены менеджером до их удаления из локального кэша.

При получении запроса на остановку передачи данных для разблокированного enabled data-req-id, агент должен прекратить отправку отчетов о событии для data-req-id immediately.

Поле data-req-id в этих отчетах о событии используется менеджером для связи этих данных измерений с соответствующим запросом данных.

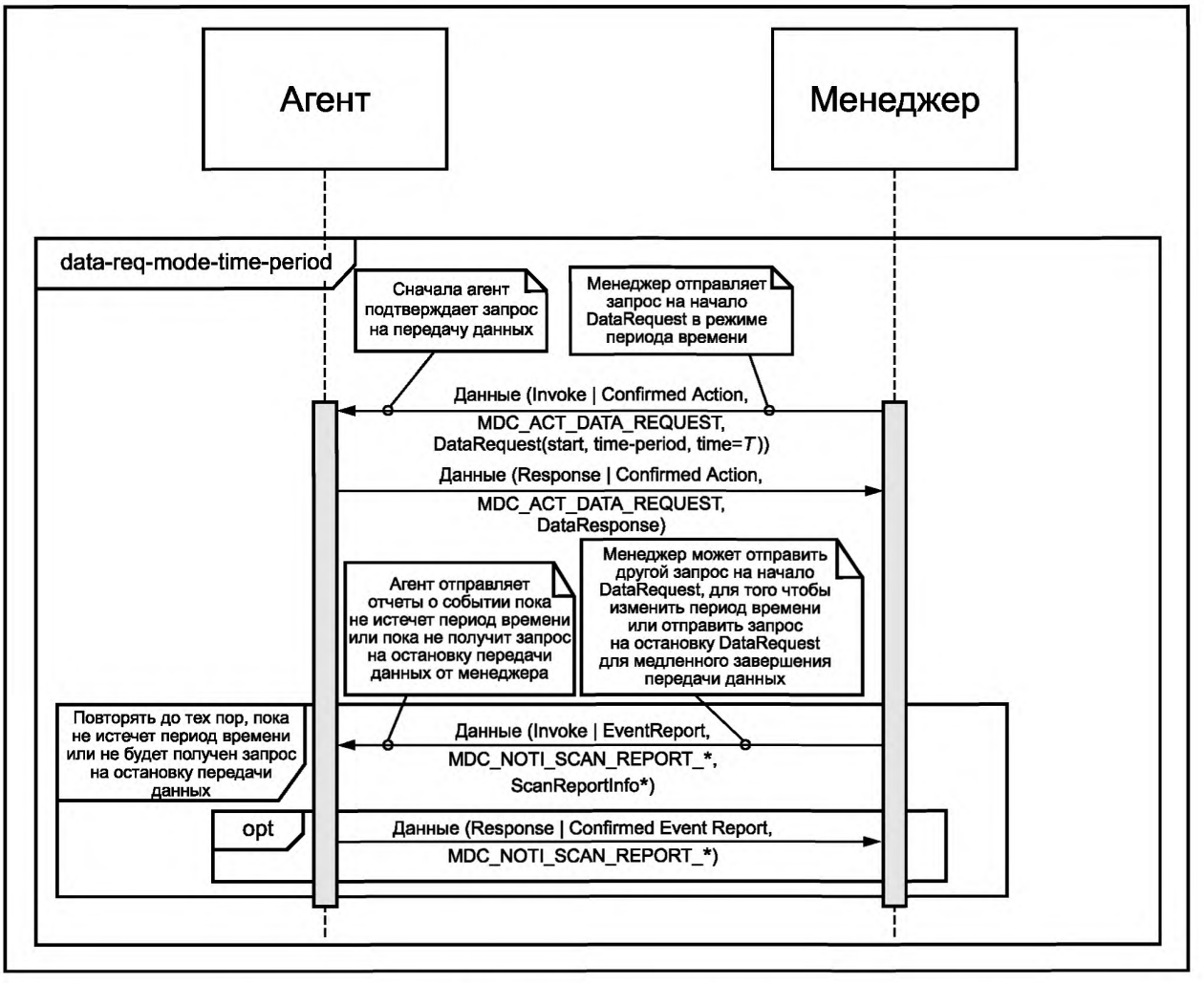


Рисунок 18 — Передача данных измерений, инициированная менеджером (data-req-mode-time-period)

8.9.3.3.6 Режим без ограничения времени, инициированный менеджером

Режим без ограничения времени должен использоваться для того, чтобы дать команду агенту отправлять отчеты о событиях постоянно до тех пор, пока не будет получена команда с запросом на остановку или ассоциация между агентом и менеджером не прекратится (см. рисунок 19).

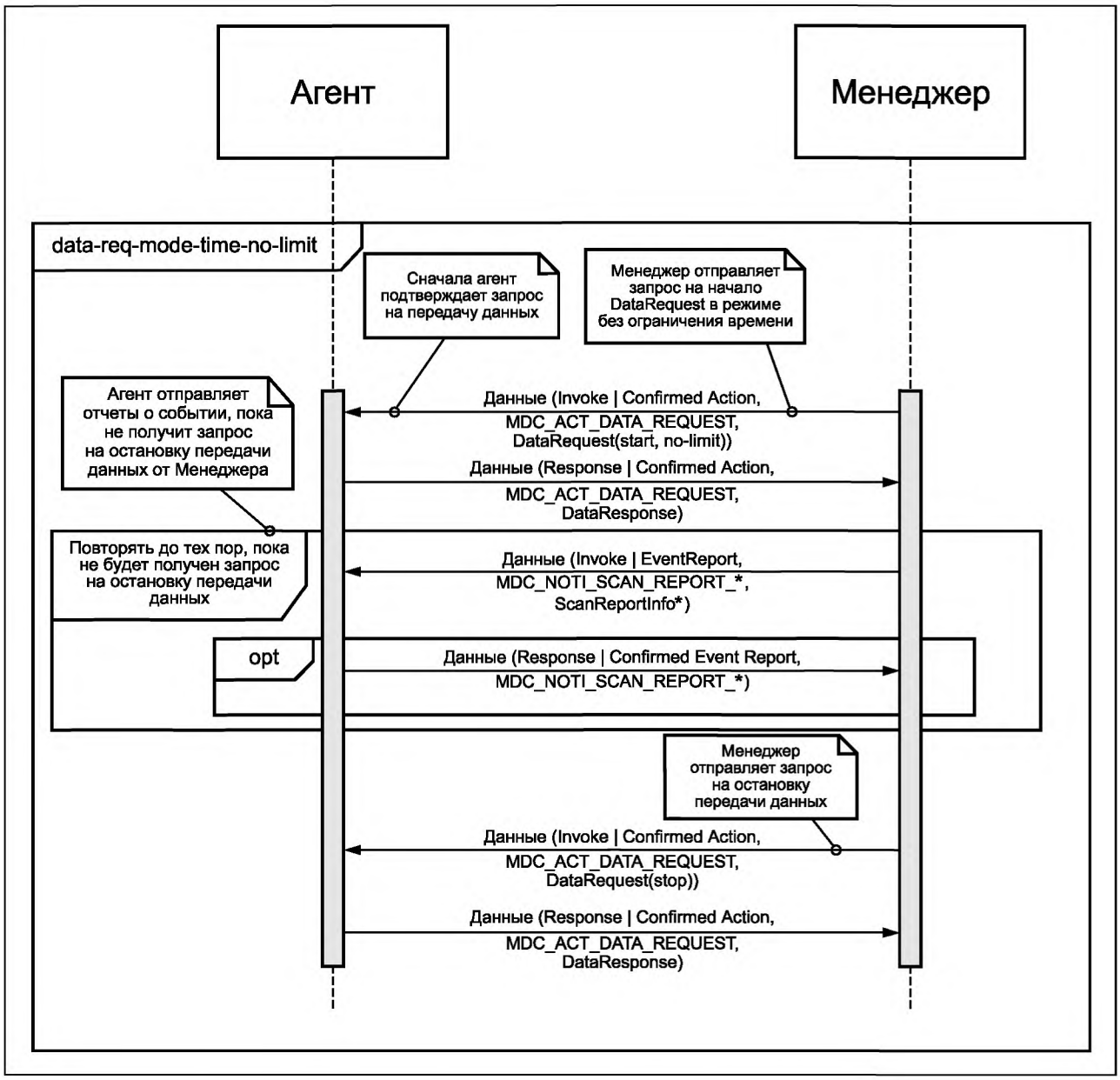


Рисунок 19 — Передача данных измерений, инициированная менеджером (data-req-mode-time-no-limit)

8.9.3.3.7 Управление номерами отчетов о сканировании

Запрос на передачу данных, в котором установлен бит data-req-start-stop, образует новый поток одного или нескольких измерительных наблюдений от агента к менеджеру в контексте системы MDS. Когда поток системы MDS образован, агент создает новый экземпляр счетчика scan-report-no для этого потока. Для каждого потока должен иметься один экземпляр счетчика scan-report-no, как дифференцированного с помощью data-req-id. Этот счетчик должен начинаться с 0 и увеличивается на 1 для каждого отчета события, отправленного на поток, увеличенного до 0. Если агент получает запрос на передачу данных, который содержит набор битов data-req-start-stop и значение data-req-id, которое уже используется в потоке системы MDS, агент не может сбросить счетчик scan-report-no идентифицированного потока до 0. Если агент получает запрос на передачу данных, который содержит набор битов data-req-continuation, scan-report-no должен продолжать подсчет без сброса.

Режим одиночного ответа, инициированный менеджером (data-req-mode-single-rsp), сформированный от передачи данных измерения, должен привести к ответу, который содержит 0 в поле scan-report-no структуры ScanReportInfo*. Это происходит, потому что новый поток создается при каждом запросе data-req-mode-single-rsp и заканчивается при отправке запроса.

Передача данных, инициированная агентом от объектов системы MDS или сканера с помощью контараста, образует поток, который заканчивается только при разрыве ассоциации. Таким образом, для передачи данных, инициированной агентом, scan-report-no начинается с 0, но не может быть сброшен менеджером в рамках ассоциации. Установка атрибута сканера Operational-State для заблокированной остановленной передачи отчетов о событиях, т.е. внутреннее наблюдение метрических объектов, останавливается и возобновляется после установки атрибута Operational-State для разблокировки. Scan-report-no в этом случае продолжает отсчет с того момента, с которого он был остановлен.

8.9.3.3.8 Множественные потоки, ссылающиеся на один объект измерения

Агент может как инициировать, так и получать запросы на потоки, которые ассоциируют data-req-ids с метрическими объектами через контекст системы MDS. Когда метрические объекты, ассоциированные с несколькими потоками, образуют данные измерений, наблюдения данных на каждом из потоков должно быть занесено в отчет.

Агент должен сообщить максимальное количество совпадающих потоков, инициированных менеджером, которые поддерживаются в data-req-init-manager-count в процессе ассоциации. Менеджер должен ограничить количество совпадающих потоков, инициированных менеджером, которые он запрашивает, для того, чтобы значение, сообщенное агентом, не было превышено. Если агент не может установить новый поток, инициированный менеджером из-за истощения ресурсов, он должен установить data-req-result дозначения data-req-result-init-manager-overflow в ответном сообщении.

8.9.3.4 Перемещение метрических данных постоянного хранения.

8.9.3.4.1 Общие положения

Когда агент вводит один или несколько объектов хранения PM, агент сообщает о существовании объекта хранения PM на этапе конфигурации. Менеджер использует эту информацию для запроса объекта(ов) хранения PM-агента. Взаимодействие между менеджером и агентом при извлечении информации из PM-блока описано в пункте 8.9.3.4.2.

8.9.3.4.2 Передача метрических данных постоянного хранения

a) Извлечение атрибутов PM-блока. Когда агент и менеджер находятся в рабочем состоянии, менеджер может проверить конфигурацию, согласованную с агентом, чтобы определить количество объектов хранения PM в агенте. Менеджер может запрашивать каждый PM-блок, чтобы определить количество сегментов PM, которые существуют в PM-блоке. На рисунке 20 показана схема последовательности этой процедуры. Менеджер отправляет команду Get (Получение) агенту с запросом о предоставлении информации атрибута из конкретного PM-блока. Менеджер использует номер дескриптора для создания ссылки на нужное хранилище PM. Менеджер должен оставить список attribute-id-list пустым для отправки запроса на возвращение всех атрибутов. Агент отправляет в ответе значения запрашиваемых атрибутов. Менеджер может изучить атрибуты, чтобы узнать конфигурацию хранилища. Например, PM-Store-Capab описывает возможности хранилища, и Number-Of-Segments определяет, сколько сегментов присутствуют в хранилище. См. таблицу 9 для получения полного списка атрибутов и их определений.

Если агент поддерживает несколько экземпляров объектов PM-блока, запрос Get требуется для каждого PM-блока.

b) Извлечение информации сегмента PM. Менеджер извлекает информацию о сегментах в PM-блоке, отправив команду ACTION (Действие). Команда Get-Segment-Info для конкретного PM-блока (см. рисунок 21) с запросом вернуть информацию из всех сегментов, из конкретного списка сегментов, или из любого сегмента в пределах заданного интервала времени. Агент должен поддерживать первые два критерия отбора и может поддерживать предел времени критериев отбора. Менеджер имеет возможность определять, обеспечивает ли агент поддержку путем проверки inspecting pmsc-abs-time-select в атрибуте информации PM-Store-Capab PM-блока, извлеченном ранее.

Агент отвечает на команду ACTION.Get-Segment-Info со списком номеров сегментов, за которым следует полный список атрибутов для каждого из сегментов.

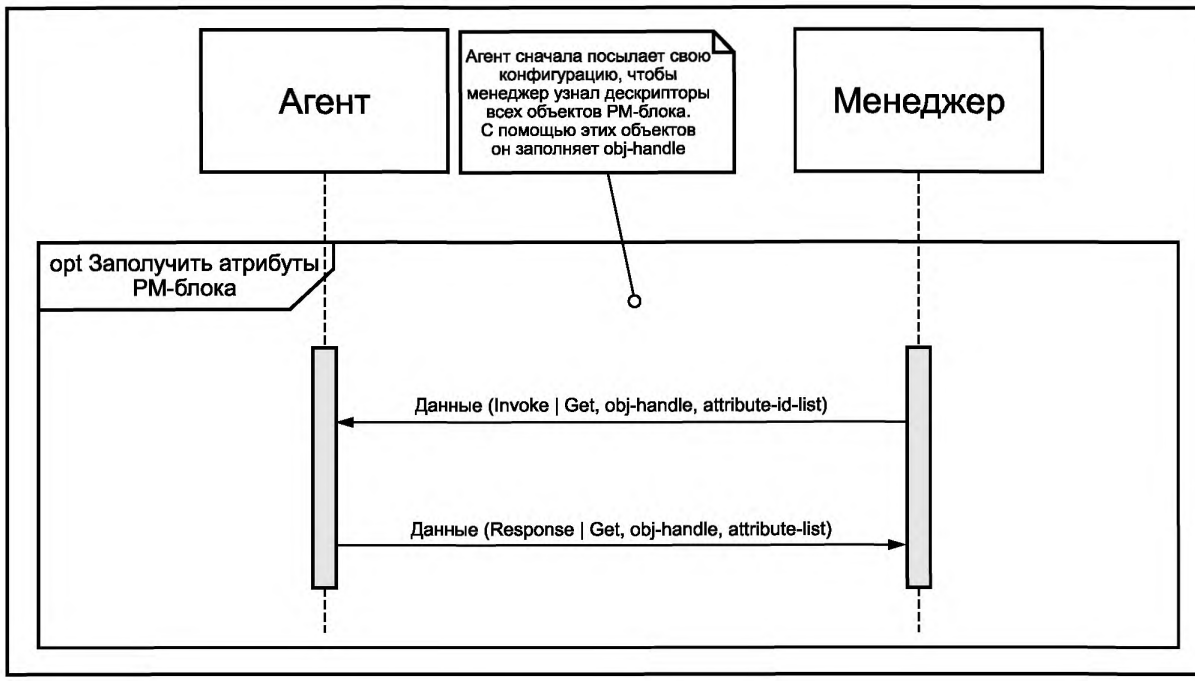


Рисунок 20 — Извлечение атрибутов РМ-блока

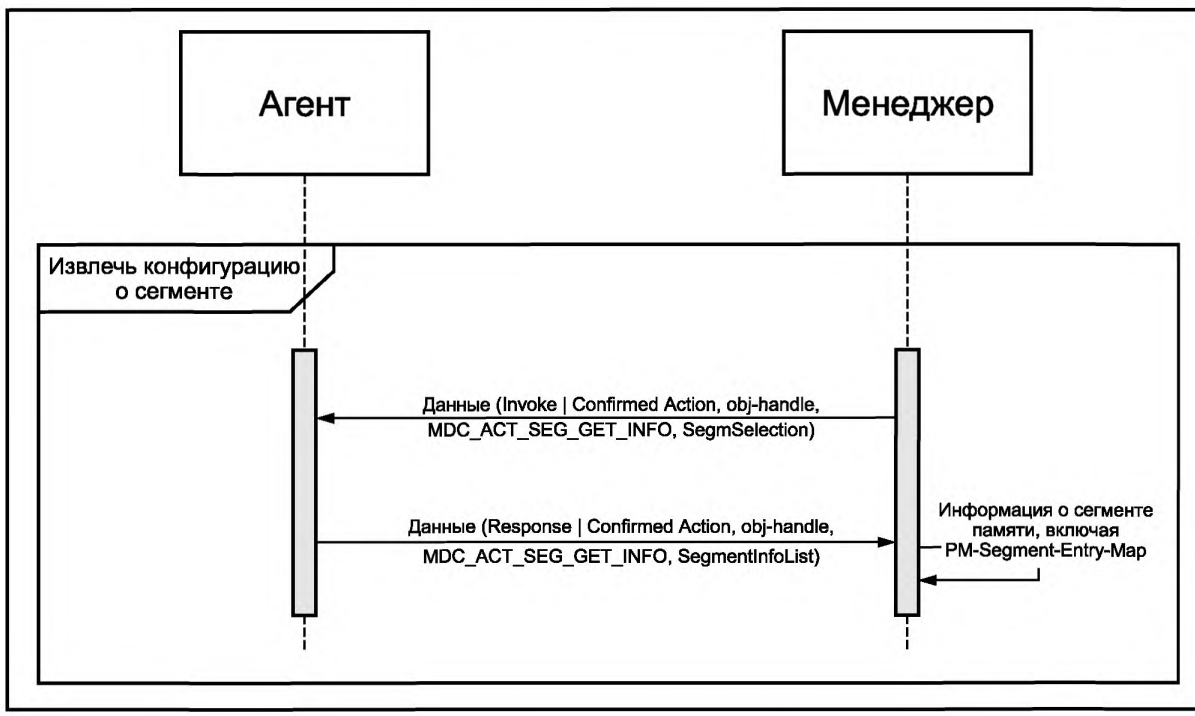


Рисунок 21 — Извлечение информации сегмента РМ

c) Перенос содержимого сегмента РМ. Менеджер получает конкретные сегменты РМ с помощью метода Trig-Segm-Data-Xfer ACTION для того, чтобы начать передачу данных (см. рисунок 22). Менеджер должен передавать информацию о дескрипторе РМ-блока для получения доступа и номера экземпляра сегмента, подлежащего передаче.

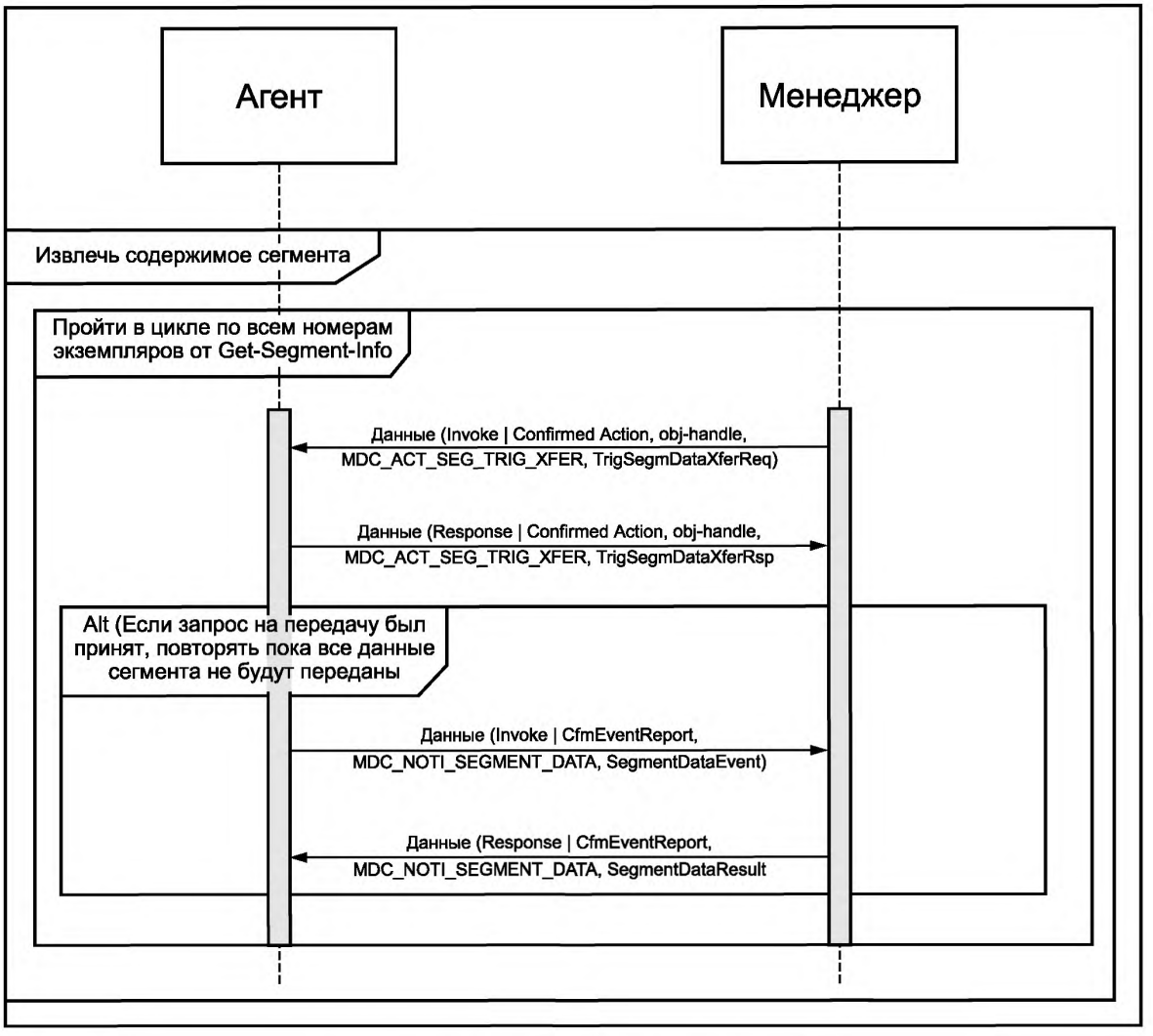


Рисунок 22 — Извлечение содержимого сегмента PM

Агент должен решить, может ли запрос быть выполнен. Он проверяет действующий номер сегмента, доступные данные о сегменте (то есть они могут быть в процессе обновления), или любые другие условия возникновения ошибок. Если возникает ошибка, агент должен сообщить соответствующий код ошибки в ответе и игнорировать запрос на передачу данных. В противном случае агент должен направить код ответа tsxr-successful, чтобы указать, что он получил запрос, и он может быть выполнен.

Менеджер может отправить сообщение о вызове Trig-Segm-Data-Xfer ACTION в любое время. Однако если менеджер отправляет сообщение о вызове Trig-Segm-Data-Xfer ACTION, в то время как сообщение о вызове Clear-Segments ACTION находится в ожидании, агент может создать ответное сообщение Clear-Segments ACTION с кодом возврата trig-segm-xfer-rsp = tsxr-fail-clearin-process. Примером того, когда этот код возврата может быть отправлен, является ситуация, при которой носитель данных для PM-блока является отдельным флеш-накопителем. Если данные флеш-накопителя стираются, это может привести к потере доступа ко всему накопителю.

Агент должен направить один подтвержденный отчет событий Segment-Data-Event, пока все записи в сегменте PM не будут отправлены менеджеру или передача не будет прервана битом sevtsta-agent-abort или битом sevtsta-manager-abort, описанным далее. Агент заполняет структуру SegmentDataEvent с информацией о сегменте, подлежащем передаче. Агент сообщает менеджеру дескриптор PM-блока

и использует SegmDataEventDescr для описания номера сегмента, подлежащего передаче, номер первой записи в поле segm-data-event-entries, количество записей в сообщении, и информацию о текущем состоянии. Агент всегда должен устанавливать любые биты sevtsta-manager-* на 0. Если сообщение содержит первую запись и/или последнюю запись записей данных, то агент должен установить бит sevtsta-first-entry и/или sevtsta-last-entry, соответственно. Если агент намерен прервать передачу, он должен установить бит sevtsta-agent-abort на 1.

При передаче сегмента агент использует поле segm-data-event-entries для отправки всех записей. Агент должен начать с первой собранной записи, за которой будет следовать следующая запись, и так далее. Для оптимизации передачи агент должен упаковать столько записей в структуру событий сколько возможно. Каждая запись должна быть отформатирована в соответствии со структурой, определенной в PM-Segment-Entry-Map сегмента PM.

Когда менеджер получает отчет о событии, то он должен ответить SegmentDataResult, который должен содержать тот же дескриптор блока (store-handle), номер экземпляра сегмента (segm-instance), segm-evt-entry-index и segm-evt-entry-count. В segm-evt-status менеджер должен установить бит sevtsta-manager-confirm.

Если агент устанавливает бит sevtsta-agent-abort, то менеджер должен подтвердить отключение агента, установив тот же бит. Если менеджер хочет прервать обмен данными, он должен установить бит sevtsta-manager-abort.

d) Очистка сегмента PM. Менеджер может очистить сегмент PM в любое время и использует последовательность, показанную на рисунке 23. Обычное время для очистки сегмента наступает непосредственно после передачи всего сегмента менеджером. Менеджер узнает это условие, когда получает SegmEvtStatus с набором битов sevtsta-last-entry.

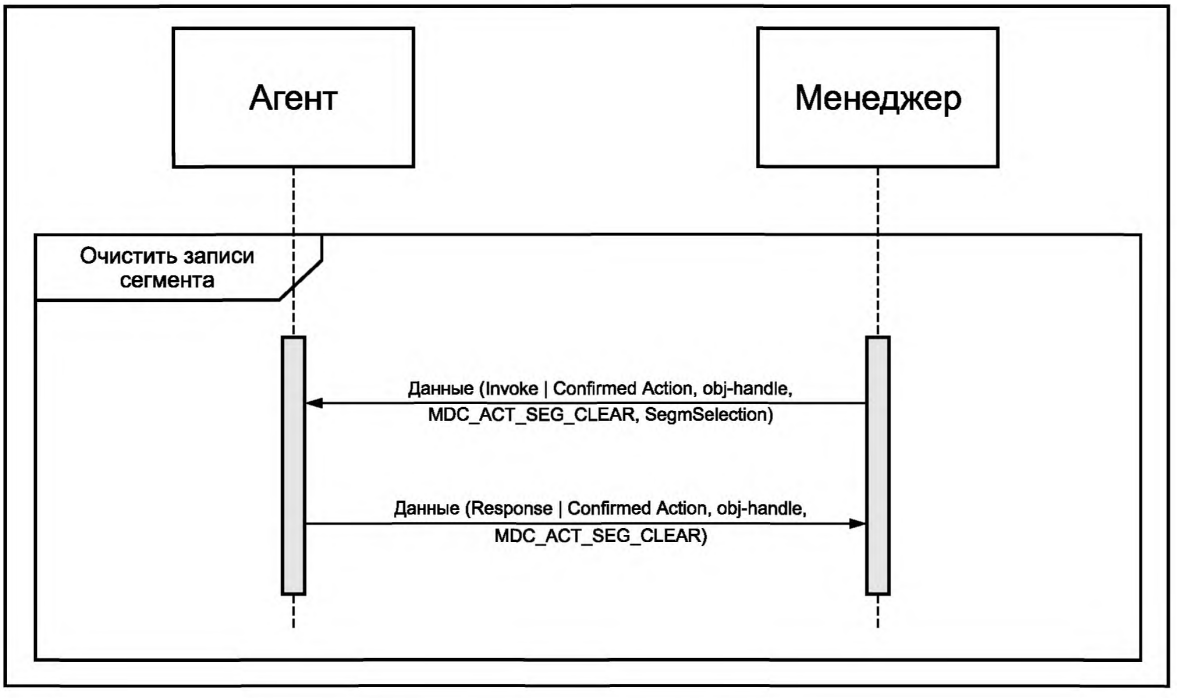


Рисунок 23 — Очистка записей сегмента

Всякий раз, когда менеджер решает очистить сегмент(ы), он посылает команду ACTION агенту с методом Clear-Segments и критериями отбора всех сегментов, конкретным списком сегментов, или любым сегментом в заданном диапазоне времени. Агент должен поддерживать очистку всех сегментов, очистку конкретного списка сегментов (pmsc-clear-segm-by-list-sup) и может поддерживать предел времени критериев отбора (pmsc-clear-segm-by-time-sup). Менеджер определяет, какие возможности поддерживаются путем проверки битов атрибута PM-Store-Capab.

Когда агент получает команду Clear-Segment, он может удалить все присутствующие записи и оставить сегмент, или может удалить сегмент. Менеджер определяет, какие возможности поддерживаются путем проверки бита pmsc-clear-segm-remove атрибута PM-Store-Capab.

8.9.4 Условия выхода

Нормальный выход из состояния Выполнения происходит тогда, когда агент или менеджер решает завершить ассоциацию. В этом случае агент или менеджер должен перейти в состояние Завершение ассоциации и следовать процедуре разъединения (см. 8.10).

Когда агент или менеджер получает запрос на завершение ассоциации, он должен отправить ответ на завершение ассоциации и перейти в неассоциированное состояние.

Агент, выходящий из состояния Выполнения, правильным или неправильным способом, должен остановить все механизмы передачи данных, включая передачу данных измерений, инициированную агентом или менеджером, передачу сегмента PM, и передачу данных со сканера.

8.9.5 Состояния ошибки

8.9.5.1 Общие положения

Как и в 8.9.4, агент, выходящий из состояния Выполнения, правильным или неправильным способом, должен остановить все механизмы передачи данных, включая запущенные агентом или менеджером передачу данных измерения, передачу PM-сегмента или передачу данных со сканера.

Если в любой момент времени происходит тайм-аут транспортного уровня от бесперебойного транспортного уровня, то агент или менеджер должны сделать следующее:

- для транспортных протоколов, зависимых от тайм-аута/подключения (например, протокол управления передачей данных) — перейти обратно к состоянию Разъединен с учетом того факта, что данные об истечении срока ожидания протокола передачи данных передаются на вышестоящие уровни как «Индикация отключения транспортного уровня»;

- для транспортных протоколов, независимых от истечения срока тайм-аута/подключения (например, USB универсальная последовательная шина) — попытаться восстановить транспортный канал, попытаться отправить сообщение о принудительном прерывании ассоциации своему партнеру, а затем перейти обратно к неассоциированному состоянию.

8.9.5.2 Сервис подтвержденного действия

После отправки сообщения для вызова сервиса подтвержденного действия менеджер должен ожидать ответного сообщения от сервиса подтвержденного действия на протяжении TO_{ca} (время работы сервиса подтвержденного действия). Если период TO_{ca} истекает, менеджер должен отправить агенту сообщение о принудительном прерывании ассоциации и перейти обратно к неассоциированному состоянию.

8.9.5.3 Сервис подтвержденного отчета о событии

После отправки сообщения для вызова сервиса подтвержденного отчета о событии агент должен ожидать ответного сообщения от сервиса подтвержденного отчета о событии на протяжении TO_{cer} (время работы сервиса подтвержденного отчета о событии). Если период TO_{cer} истекает, агент должен отправить менеджеру сообщение о принудительном прерывании ассоциации и перейти обратно к неассоциированному состоянию.

Период TO_{cer} определяется отдельно для каждого объекта. Каждый из объектов в настоящем стандарте, создающий отчеты о событии, имеет свое значение времени работы, сообщаемое соответствующим атрибутом в каждом объекте:

- $TO_{cer\text{-}mds}$ (TO_{cer} для объекта системы медицинского прибора (MDS)) MDS.Confirmed-Timeout;
- $TO_{cer\text{-}pms}$ (TO_{cer} для объекта PM-блока) Segm.Confirmed-Timeout;
- $TO_{cer\text{-}scan}$ (TO_{cer} для объекта сканера) Scan.Confirmed-Timeout.

8.9.5.4 Сервис получения (Get)

После отправки сообщения для вызова Get менеджер должен дождаться ответного сообщения за период TO_{get} (время работы получения сервиса). Если период TO_{get} истекает, то менеджер должен отправить агенту сообщение о принудительном прерывании ассоциации и перейти обратно к неассоциированному состоянию.

8.9.5.5 Сервис подтвержденной установки

После отправки сообщения для вызова сервиса подтвержденной установки менеджер должен ожидать ответного сообщения о сервисе подтвержденной установки на протяжении периода TO_{cs} (время работы сервиса подтвержденной установки). Если период TO_{cs} истекает, то менеджер должен отправить агенту сообщение о принудительном прерывании ассоциации и перейти обратно к неассоциированному состоянию.

8.9.5.6 Специальные тайм-ауты

В дополнение к обычным тайм-аутам сервиса связи, описанным выше, существует три тайм-аута для частных случаев, которые также используются в протоколах персональных медицинских приборов:

$TO_{clr-pms}$ — специальное время работы для очистки объекта хранения PM-блока (PMS.Clear-Timeout);

TO_{sp-mds} — специальный период времени между сервисами для объектов MDS (3 с);

TO_{sp-pms} — специальный период времени передачи сегмента PM-блока (Segm.Transfer-Timeout).

$TO_{clr-pms}$. После отправки сообщения вызова сервиса подтвержденного действия (MDC_ACT_SEG_CLR) менеджер должен ожидать ответного сообщения от сервиса подтвержденного действия на протяжении периода $TO_{clr-pms}$ (время работы сервиса подтвержденного действия для очистки объекта хранения PM). Если период $TO_{clr-pms}$ истекает, то менеджер должен отправить агенту сообщение о принудительном прерывании ассоциации и перейти обратно к неассоциированному состоянию.

TO_{sp-mds}. После отправки сообщения вызова сервиса подтвержденного действия (MDC_ACT_DATA_REQUEST, старт, промежуток времени, время = 0) менеджер должен ожидать сообщение для вызова сервиса подтвержденного отчета о событии на протяжении TO_{sp-mds} (специальный период времени между сервисами для объектов MDS). Если период TO_{sp-mds} истекает, то менеджер должен отправить агенту сообщение о принудительном прерывании ассоциации и перейти обратно к неассоциированному состоянию.

TO_{sp-pms}. После отправки сообщения для вызова сервиса подтвержденного действия (MDC_ACT_SEG_TRIG_XFER) менеджер должен ожидать сообщения вызова сервиса подтвержденного отчета о событии (segm-evt-status=sevtsta-last-entry, semg-data-event-entries) на протяжении периода TO_{sp-pms} (специальный период времени передачи сегмента PM-блока). Если период TO_{sp-pms} истекает, менеджер должен отправить агенту сообщение о принудительном прерывании ассоциации и перейти обратно к неассоциированному состоянию.

8.10 Процедура Завершения ассоциации

8.10.1 Общие положения

Процедура завершения ассоциации предоставляет любому агенту или менеджеру механизм, позволяющий правильно завершить ассоциацию.

8.10.2 Условия входа

Когда агент или менеджер решают закрыть ассоциацию, они должны перейти обратно к неассоциированному состоянию и запустить процедуру завершения ассоциации.

8.10.3 Обычные процедуры

В неассоциированном состоянии агент отправляет запрос на завершение ассоциации своему партнеру и дожидается ответа. Запрос на завершение ассоциации содержит причину завершения ассоциации (ReleaseRequestReason):

- причина отсутствия конфигураций используется агентом в состоянии Конфигурации для указания того, что все возможные конфигурации были испробованы и отклонены менеджером;
- причина изменения конфигурации используется агентом в состоянии Выполнения для указания того, что конфигурация менеджера была изменена и больше нет возможности отправлять данные с прежней согласованной конфигурацией. Обычно агент отвечает на это сообщение новым запросом на ассоциацию с новым dev-config-id; однако данный шаг не является обязательным;
- обычная причина используется агентом или менеджером для выхода из состояния Выполнения без указания особого условия.

Если агент или менеджер получают запрос на завершение ассоциации с нестандартным идентификатором вызова, они должны отправить Ответ на запрос на завершение ассоциации и считать, что ответа на запрос не последует.

8.10.4 Условия выхода

Когда агент или менеджер получают ответ на запрос на завершение ассоциации, они должны перейти к неассоциированному состоянию.

Если агент или менеджер получают Запрос на завершение ассоциации в неассоциированном состоянии, то они должны отправить Ответ на запрос на завершение ассоциации и оставаться в неассоциированном состоянии и ждать ответа на собственный Запрос на завершение ассоциации.

8.10.5 Условия ошибки

После отправки сообщения о завершении ассоциации агент или менеджер должны ожидать Ответа на запрос о завершении ассоциации на протяжении периода $TO_{release}$ (время работы процедуры завершения ассоциации). Если период $TO_{release}$ истекает, менеджер должен отправить агенту сообщение о принудительном прерывании ассоциации своему партнеру и перейти обратно к неассоциированному состоянию.

Агент или менеджер могут отправлять или получать сообщение о принудительном прерывании ассоциации для других неисправных состояний и должны немедленно перейти к неассоциированному состоянию.

8.11 Кодирование сообщений

Используемый в настоящем стандарте язык ASN.1 для описания абстрактного синтаксиса поддерживает конвертирование во множество других форматов передачи. Менеджер и агент должны поддерживать правила кодирования медицинских приборов (MDER), как указано в стандарте ИСО/ИИЭР 11073-20101:2004 [14]. Правила кодирования MDER представлены в приложении F наряду с дополнительными оптимизациями, специфичными для настоящего стандарта. В дальнейшем для двоичных передач должен быть использован порядок передачи данных (кодирование с обратным порядком байтов). Настоящий стандарт также допускает возможность агенту или менеджеру использования правил уплотненного кодирования (ASN.1) (PER) [17] и правила кодировки расширяемого языка разметки (XML) (XER) [18].

В приложении G дан один из примеров того, как структура данных языка ASN.1 может быть перекодирована в синтаксис языка С.

Приложение Н содержит вспомогательные примеры бинарных кодировок, получающихся из сообщений, обозначенных в настоящем стандарте.

Все номенклатурные коды, использованные в настоящем стандарте, определены с использованием MDC_.. представления, но при передаче должны быть использованы номенклатурные коды. Приложение I содержит список определенных значений для всех кодов, использованных в настоящем стандарте.

8.12 Координация времени

8.12.1 Общие положения

Агент может использовать три типа часов: часы абсолютного времени, часы относительного времени и часы относительного времени высокой точности. Во всех случаях информацию о возможностях часов агента и о том, синхронизированы ли одни или несколько часов с внешним источником времени, можно найти через атрибут Mds-Time-Info в таблице 2. Все ссылки на биты в подразделах являются частью данного атрибута. Все агенты с любым типом часов должны поддерживать данный атрибут.

8.12.2 Абсолютное время

8.12.2.1 Общая информация

Агенты с внутренними часами реального времени (RTC) должны отображать данную возможность, устанавливая бит mds-time-capab-real-time-clock (см. А.11.1). Агенты, которые поддерживают действие установки часов (см. 6.3.2.4 и А.4), должны устанавливать бит mds-time-capab-set-clock.

Агенты могут поддерживать независимый способ для синхронизации внутренних часов реального времени с внешним источником часов. Используемый метод синхронизации не входит в область применения настоящего стандарта. Однако агент должен указывать, синхронизируется ли абсолютное время, используя бит mds-time-capab-sync-abs-time. Если синхронизация поддерживается, протокол, используемый для синхронизации внутренних часов реального времени (например, сетевой протокол синхронизации времени и простой сетевой протокол синхронизации времени), сообщается в разделе протокола синхронизации времени с использованием таких идентификаторов как MDC_EXT_PROTO_TIME_NTP. Бит mds-time-state-abs-time-synced должен быть установлен только тогда, когда агент считает, что его атрибуты даты и времени синхронизированы с внешним источником часов.

У агентов может появиться необходимость указать менеджеру, должен ли он установить время посредством действия Set-time. Если бит mds-time-mgr-set-time установлен, менеджер должен использовать команду действия Set-time, чтобы установить агенту абсолютное время. Если не установлен, то агент не хочет, чтобы менеджер установил часы. Такая ситуация может возникнуть, когда агент син-

хронизирует часы через внешний источник часов или когда пользователь установил часы на местном уровне. В этом случае менеджер не должен пытаться установить часы.

Такие атрибуты, как Date-and-Time и Absolute-Time-Stamp, сообщают дату и время агента. Для некоторых областей применения важно, чтобы агент сообщал дату и время, которые были показаны лицу, пользующемуся устройством (например, на глюкометре). Для других областей применения необходимо сообщать время, которое координируется с системой единого времени, таким, как универсальное координированное время (UTC). Например, такая ситуация может возникнуть, когда дата и время установлены на заводе по UTC и устройство не предусматривает средств для изменения даты и времени. Со стороны менеджера способность связать время измерения с понятием менеджера о времени имеет решающее значение для некоторых областей применения.

8.12.2.2 Сопоставимое время

В данном стандарте для всех трех областей применения используется понятие «сопоставимого времени». Основными понятиями сопоставимого времени являются следующие:

- когда агент сообщает информацию о времени, она должна обеспечить, что все указанные в комплекте измерения входят в одну непрерывную временную шкалу. Для временных измерений комплект состоит из всех измерений в одном отчете событий. Для PM-блока набор эквивалентен PM-сегменту;

- если набор измерений был собран, когда текущие часы имели другие настройки, то агент должен либо удалить данные, либо сообщить данные наряду с количеством 1/100 секунд, которые необходимо добавить к каждому из времен измерения, чтобы разместить их на той же временной шкале, что и текущие атрибуты Date-and-Time агента;

- эти два понятия применяются только в случае, если временная ось, использованная для отметки значений времени, изменилась на значение, значимое для типа измерений. Другими словами, нет необходимости сообщать о небольших смещениях времени или незначительном изменении настроек часов, чтобы поддерживать синхронизацию с внешним источником времени.

Абсолютное время должно интерпретироваться как сопоставимое время для агента следующим образом:

- если агент связан с менеджером во время настроек атрибутов Date-and-Time, то он должен отправить отчет о событиях, содержащий новое значение атрибутов Date-and-Time. Единственным исключением является случай, когда для изменения времени агента менеджер использует команду Set-Time. В этом случае агент может принять решение не посылать отчет о событиях, так как менеджер уже знает об изменении времени;

- если агент собирает временные измерения, а атрибуты Set-Time настроены, агент должен обеспечить, чтобы все измерения, включенные в отчет о событиях, входили в одну непрерывную временную шкалу, то есть не было внесено никаких изменений в настройки интервалов временных меток, включенных в данный отчет о событиях. Кроме того, все отчеты о событиях, которые содержат измерения, произведенные до того времени, когда было настроено текущее время агента, должны иметь атрибут MDS Date-and-Time-Adjustment в качестве первых указанных данных в отчете о событиях. Этот атрибут должен указывать количество 1/100 секунд, которые необходимо добавить к каждой временно́й метке в протоколе о событиях для согласования с текущими часами (например, если часы были переведены на 60 минут вперед, в отчете это было бы обозначено как 360 000);

- если агент собирает измерения PM-блока, а атрибуты даты и времени настроены, агент должен обеспечить, чтобы каждый PM-сегмент включал только измерения с одной непрерывной временно́й шкалы. Кроме того, в каждом PM-сегменте должен присутствовать атрибут Date-and-Time-Adjustment PM-сегмента, который содержит измерения, собранные с учетом других настроек часов;

- следует отметить, что в тех случаях, когда измерения собраны в автономном режиме, если настройки часов были изменены несколько раз, перед загрузкой данных, значение Date-and-Time-Adjustment является суммарным. Другими словами, сначала собираются измерения, затем часы переводятся назад на 30 мин, собираются дополнительные измерения, а часы переводятся назад еще на 30 мин. В этом случае в первом наборе данных должно быть указано смещение на 60 мин, а во втором набор указывается смещение на 30 мин.

8.12.3 Относительное время

Агенты могут применять относительный таймер с разрешающей способностью до 125 мкс [наименее значащий двоичный бит (LSB)]. Данное разрешение является достаточным для частоты выборки до 8 кГц, позволяет измерять периоды относительного времени с высокой точностью и охватывает периоды времени до 6,2 дня. Если относительное время используется с временно хранящимися измерениями или PM-блоком, агенты должны гарантировать, что продолжительность времени хранения

никогда не превышает разрешающую способность таймера (т. е. 6,2 дня). Такая гарантия со стороны агента позволяет менеджеру запрашивать текущее относительное время агента и вычислять, как давно были произведены измерения. Если требуется большее время хранения, то используются либо атрибуты абсолютного времени, либо атрибуты относительного времени с высокой точностью. Агенты должны указывать поддержку относительного времени посредством установки бита mds-time-capab-relative-time в атрибуте Mds-Time-Info. Этот таймер должен быть запущен перед ассоциацией. За исключением случаев перезапуска счетчика, он должен однообразно увеличивать счет и не менять свое значение после инициализации. Точность фактического времени (то есть внутренний период обновления) определяется агентом, но должна соответствовать цели данного устройства.

Агенты могут поддерживать способ синхронизации своего внутреннего таймера с внешним источником часов (например, через пикосеть Bluetooth). Используемый метод синхронизации не входит в область применения настоящего стандарта. Однако агент должен указывать, синхронизирует ли он относительное время с использованием бита mds-time-capab-sync-rel-time. Если синхронизация поддерживается, то бит mds-time-state-rel-time-synced должен быть установлен только тогда, когда агент считает, что его часы относительного времени синхронизируются с внешним источником. Все агенты, связанные с одним менеджером и указывающие, что их внутренние таймеры синхронизированы, должны показывать одно и то же относительное время для событий, синхронизированных во времени.

Если агент предоставляет метку относительного времени для численного измерения, временная метка должна находиться точно в пределах точности синхронизации установленного времени и времени выборки численного значения. Метка относительного времени при использовании в качестве времени события и текущего времени может предоставить точную информацию об интервалах между событиями. Метка относительного времени может обеспечить точные измерения абсолютного времени, когда менеджер получает атрибуты относительного и абсолютного времени от объекта системы MDS агента и определяет время относительно собственных внутренних часов.

Если агент предоставляет метку относительного времени для массива показаний, снятых в режиме реального времени (RT-SA), то метка времени должна относиться к первой выборке в данном массиве и временная метка должна находиться точно в пределах заявленной точности атрибута относительного времени и времени выборки RT-SA.

Отчеты о событиях могут содержать метки относительного времени, указывающие, когда было создано событие. Если метрически производные объекты, содержащиеся в протоколе событий, не имеют основную временную метку, то время события будет также использовано в качестве времени измерения. Если агент предоставляет метки относительного времени для времени события, временная метка находится точно в пределах заявленной точности атрибута относительного времени, времени события, а также любых атрибутов времени выборки, связанных с событием.

8.12.4 Относительное время высокой точности

Агенты могут применять внутренний таймер высокой точности с разрешающей способностью до 1 мкс (LSB). Данный таймер высокой точности обладает точностью, достаточной для частоты выборки до 1 МГц, позволяет измерять периоды относительного времени с высокой точностью и охватывает периоды времени до 584 000 лет. Агенты должны указывать поддержку данного свойства посредством установки бита mds mds-time-capab-high-res-relative-time в атрибуте Mds-Time-Info. Этот таймер должен быть запущен перед ассоциацией. За исключением случаев перезапуска счетчика, он должен однообразно увеличивать счет и не менять свое значение после инициализации. Необходимая разрешающая способность (то есть внутренний период обновления) определяется агентом, но должна соответствовать цели данного устройства. Относительное время высокого разрешения должно сохранять частотную синхронизацию с относительным временем.

Агенты могут поддерживать способ синхронизации своего внутреннего таймера высокой точности с внешним источником часов (например, через пикосеть Bluetooth). Используемый метод синхронизации не входит в область применения настоящего стандарта. Однако агент должен указывать, синхронизирует ли он относительное время высокой точности с использованием бита mds-time-capab-sync-hi-res-relative-time. Если синхронизация поддерживается, то бит mds-time-state-hi-res-relative-time-synced должен быть установлен только тогда, когда агент считает, что его часы относительного времени синхронизируются с внешним источником. Когда агент отключается от источника синхронизации часов, он должен обозначить бит синхронизации как «ложное значение», как только он превысит точность параметров синхронизации часов.

Если агент обеспечивает метку относительного времени высокой точности для численного измерения, временная метка должна находиться точно в пределах заявленной точности атрибута относительного времени и времени выборки числового значения.

Если агент предоставляет метку относительного времени высокой точности для RT-SA, метка времени должна относиться к первой выборке в данном массиве. Временная метка должна находиться точно в пределах заявленной точности атрибута относительного времени и времени выборки RT-SA.

9 Модель соответствия

9.1 Применимость

Настоящий стандарт предполагается использовать совместно с другими базовыми стандартами или в качестве ссылки в других базовых стандартах семейства ИСО/ИИЭР 11073 для определения приложений (например, для обмена базами данных измерений показателей жизненно важных функций) или для определения функциональных профилей коммуникаций (например, профилей интероперабельности медицинских приборов).

В частности, ряд специализаций устройств ИСО/ИИЭР 11073-104zz необходим для обеспечения интероперабельности системы. Таким образом, интероперабельность требует проверки на соответствие с настоящим стандартом и рядом применяемых специализаций устройств. Специализации устройств определяют подходящую модель соответствия, которая включает в себя требования соответствия настоящему стандарту, связанные с представлением показателей жизненно важных функций. Специализации устройств используют информацию, специфическую для устройства, при определении дополнительных критериев соответствия, которые не входят в область применения настоящего стандарта.

Соответствие определениям настоящего стандарта указывается только для интерфейсов соответствующего приложения или системы. Кроме того, поведение, указанное в настоящем стандарте (например, соблюдение принципов работы определенных машин состояния), также является частью спецификации. Поведение на этом уровне считается критически важным для соответствия, чтобы обеспечить надлежащую и точную работу протокола в целом. Спецификации соответствия не распространяются на детали реализации, такие как язык программирования, выделение уровней программного обеспечения, внутренние интерфейсы и так далее.

9.2 Спецификация соответствия

Настоящий стандарт, посвященный представлению показателей жизненно важных функций, обладает высокой степенью гибкости в применении описанной в нем модели для определенного медицинского прибора, в особенности для следующих областей:

- построение информационной модели конкретного прибора;
- использование атрибутов, диапазонов значений и доступа;
- использование дополнительных (расширенных) сервисов коммуникаций (т. е. сканнеров), периодов сканирования и конфигурируемости сканеров.

Для обеспечения интероперабельности приложений и систем реализация, основанная на настоящем стандарте, должна сопровождаться специфическими подробностями того, каким образом применяются определения настоящего стандарта в сочетании с требованиями соответствия любыми производными специализациями устройств.

Подобные спецификации должны быть предоставлены в форме набора деклараций о соответствии реализации (ICS). ICS представляет собой разновидность перечня данных, в котором раскрываются подробности определенной реализации и указывается, какие свойства она предоставляет. Определенные приложения или функциональные профили коммуникаций, основанные на настоящем стандарте, должны определять более конкретные требования к соответствию в дополнение к или в качестве замены деклараций ICS, определенных в настоящем стандарте.

Примечание — Декларации соответствия, определенные в 10.3, служат для ознакомления с деталями реализации. Тем не менее их одних недостаточно для обеспечения интероперабельности приборов и приложений. Для подобной интероперабельности следует учитывать дополнительные спецификации (например, длительность, задержки, предположения о нагрузке системы). Данные спецификации не входят в область применения настоящего стандарта.

9.3 Декларации о соответствии реализации (ICS)

Декларации ICS должны быть представлены в форме таблиц. Шаблоны для подобных таблиц ICS представлены в таблицах 22—29. Таблицы должны быть заполнены и предоставлены в качестве полного документа декларации о соответствии. Как правило, заголовки столбцов ICS таблиц содержат следующую информацию:

- индекс, являющийся идентификатором (например, номером) определенного свойства;
- свойство, т. е. краткое описание характеристики, для которой должно быть сделано заявление о соответствии;
- ссылка, т. е. ссылка на определение свойства (может быть не заполнено);
- статус, устанавливающий требования соответствия (т. е. требования для претендующей на соответствие реализации, связанные с рассматриваемым свойством). Для некоторых случаев настоящий стандарт не указывает требования соответствия, но, так или иначе, требует определения статуса конкретного свойства;
- поддержка, которая заполняется специалистом по внедрению (реализации) и содержит характеристики свойств реализации;
- комментарий, содержащий дополнительную информацию, предоставленную специалистом по внедрению.

9.4 Общее соответствие

Таблицы 22—24 предназначены для использования при описании общего основного соответствия настоящему стандарту. В 9.4.1—9.4.3 сформулированы фундаментальные аспекты, обеспечивающие для устройства требования соответствия.

9.4.1 Общая декларация о соответствии реализации (ICS)

В общей ICS наивысшего уровня специалист по внедрению указывает версии/исполнения, поддерживаемые реализацией, а также некоторые определения высокого уровня для поведения системы.

В таблице 22 приведена общая ICS.

Таблица 22 — Общая ICS

Индекс	Свойство	Ссылка	Статус	Поддержка	Комментарий
GEN-1	Описание реализации	—	Идентификация устройства/приложения. Описание функционала		
GEN-2	Стандартная версия документа	(Стандартные документы)	Обозначение поддерживаемых версий по стандарту ИИЭР 11073-20601	(Набор поддерживаемых версий ИИЭР 11073-20601)	
GEN-3	Соблюдение соответствия -Уровень 1-	—	Основная декларация о соответствии устройств следующим требованиям соответствия ИИЭР 11073-20601: A) все минимальные обязательные (должны быть соблюдены) требования (см. таблицу 23 в п. 9.4.2 для некоторых из наиболее важных аспектов); B) все условные элементы были применены в соответствии с указанными условиями; C) все применяемые дополнительные элементы определяются как части декларации о соответствии (например, в таблицах ICS с Атрибутами (см. таблицу 27))	Да/Нет («Нет» подразумевает несоответствие)	

Окончание таблицы 22

Индекс	Свойство	Ссылка	Статус	Поддержка	Комментарий
GEN-4	Соблюдение соответствия -Уровень 2-	—	Вдобавок к GEN-3 устройство соответствует одной или нескольким специализациям устройства, основанным на стандарте ИИЭР 11073-20601	(Перечислите ряд специализаций устройства по ИИЭР 11073-20601, которые были реализованы, и подготовьте информацию, указанную в 9.5)	
GEN-5	Профиль связи и аппаратного обеспечения	—	Описание требований к инфраструктуре связи и аппаратному обеспечению для интерфейса		

Для каждой реализации должна быть предоставлена одна общая ICS.

9.4.2 Минимальные требования ICS

В таблице 23 показаны минимальные требования к соответствию настоящему стандарту.

Таблица 23 — Минимальные требования ИИЭР 11073-20601

Индекс	Свойство	Ссылка	Статус	Поддержка	Комментарий
REQ-1	Конечный автомат	—	-Обязательный- Имеет ли реализация персональных медицинских приборов строгое соответствие структурированному поведению конечного автомата согласно стандарту ИИЭР 11073-20601?	Да/Нет («Нет» подразумевает несоответствие)	
REQ-2	Протокольные сообщения	—	-Обязательный- Соответствует ли реализация сообщениям протокола персональных медицинских приборов согласно стандарту ИИЭР 11073-20601?	Да/Нет («Нет» подразумевает несоответствие)	
REQ-3	Объекты	—	-Рекомендованный- Соответствуют ли все объекты стандарту ИИЭР 11073-20601 или специализациям устройства, основанным на стандарте ИИЭР 11073-20601? Настоятельно рекомендуется соответствие данному набору объектов, областей, значений и поведений	Да/Нет. Если «Нет», перечислите расширения, как описано в 9.5.2	
REQ-4	Кодирование	—	-Обязательный- Поддерживаются ли Правила кодирования медицинских приборов (MDER)? Сообщения протокола кодируются из описания на языке ASN.1 в/из формат передачи данных с использованием правил кодирования. MDER должны поддерживаться. Данные правила кодирования указаны в приложении F стандарта ИИЭР 11073-20601. Существует возможность использования альтернативных правил кодирования. Перечислить все поддерживаемые правила кодирования	Да/Нет («Нет» подразумевает несоответствие) (Список поддерживаемых альтернативных правил кодировки)	
REQ-5	Спецификация	—	-Обязательный- Стандарт ИИЭР 11073-20601 основан на номенклатуре ИСО/ИИЭР 11073-10101 [12] для основной номенклатуры. Стандарт ИИЭР 11073-20601 и связанные с ним специализации устройства вносят в него дополнения. Соответствуют ли все номенклатурные коды одному из этих источников?	Да/Нет («Нет» подразумевает несоответствие)	

Окончание таблицы 23

Индекс	Свойство	Ссылка	Статус	Поддержка	Комментарий
REQ-6	Передача	—	-Обязательный- Перечислите все транспортные классы (такие как надежные и/или лучшие из возможных), которые поддерживаются реализацией. Выполнены ли для этих транспортных классов все требования передачи, как указано в ИИЭР 11073-20601?	(Список классов передачи) Да/Нет («Нет» подразумевает несоответствие)	

9.4.3 Декларация о соответствии реализации для поддержки сервисов

ICS для поддержки сервисов определяет, какие из сервисов, заданных в сервисной модели, реализованы. Подобная ICS необходима только для коммуницирующих приборов.

ICS для поддержки сервисов приведена в таблице 24.

Таблица 24 — Минимальные требования ИИЭР 11073-20601

Индекс	Свойство	Ссылка	Статус	Поддержка	Комментарий
SRV-1	Сервис ПОЛУЧИТЬ СЕРВИС (GET service)	7.3	Поддерживает ли реализация ПОЛУЧИТЬ СЕРВИС? Условный	Отправляет команду и/или получает команду, или не поддерживается	
SRV-2	Сервис УСТАНОВКИ (SET)	7.3	Поддерживает ли реализация Сервис УСТАНОВКИ? Условный	Отправляет команду и/или получает команду, или не поддерживается	
SRV-3	Сервис подтвержденной УСТАНОВКИ	7.3	Поддерживает ли реализация Сервис подтвержденной УСТАНОВКИ? Необязательный	Отправляет команду и/или получает команду, или не поддерживается	
SRV-4	Сервис ОТЧЕТА О СОБЫТИИ	7.3	Поддерживает ли реализация Сервис ОТЧЕТА О СОБЫТИИ? Условный	Отправляет команду и/или получает команду, или не поддерживается	
SRV-5	Сервис подтвержденного ОТЧЕТА О СОБЫТИИ	7.3	Поддерживает ли реализация Сервис подтвержденного ОТЧЕТА О СОБЫТИИ? Условный	Отправляет команду и/или получает команду, или не поддерживается	
SRV-6	Сервис ДЕЙСТВИЯ	7.3	Поддерживает ли реализация Сервис ДЕЙСТВИЯ? Условный	Отправляет команду и/или получает команду, или не поддерживается	
SRV-7	Сервис подтвержденного ДЕЙСТВИЯ	7.3	Поддерживает ли реализация Сервис подтвержденного ДЕЙСТВИЯ? Необязательный	Отправляет команду и/или получает команду, или не поддерживается	

Столбец Поддержка заполненной таблицы должен быть определен, если реализация вызывает сервис (например, отправляет GET PDU), предоставляет сервис (например, обрабатывает полученный GET PDU) или не реализует сервис вообще. Помимо этого, должны быть перечислены специфические ограничения (например, если определенный сервис ограничен одним классом объекта).

9.5 Дополнения к устройствам/расширения ICS

Таблицы 25—29 предназначены для использования при описании ICS для любых дополнений или расширений, используемых устройством за пределами области применения настоящего стандарта и

его специализации. Ожидается, что все условные или необязательные поведения прописаны как часть соответствующей декларации о соответствии для соответствующих специализаций устройства.

9.5.1 Общие дополнения/расширения ICS

Общие дополнения/расширения ICS определяют основную предварительную информацию по объему поддерживаемых дополнений/расширений.

Таблица 25 — Общие дополнения/расширения ICS

Индекс	Свойство	Ссылка	Статус	Поддержка	Комментарий
ADD-1	Использование частных объектов	—	Используются ли в реализации объекты, не указанные в стандарте ИИЭР 11073-20601 или одной из перечисленных специализаций устройства?	Да/Нет [Если «Да»: ICS объекта и класса информационной модели предметной области здоровья (POC) (см. 9.5.2) должно быть использовано для разъяснения подробностей реализации]	
ADD-2	Использование кодов, не входящих в номенклатуру 20601 из стандарта ИСО/ИИЭР 11073-10101 [12]	—	Используются ли в реализации номенклатурные коды из стандарта ИСО/ИИЭР 11073-10101 [12], не указанные в стандарте ИИЭР 11073-20601 или одной из перечисленных специализаций устройства?	Да/Нет (Если «Да»: объясните в подходящем ICS, см. 9.5.6)	
ADD-3	Использование частных номенклатурных расширений	—	Используются ли в реализации частные расширения для номенклатуры? Частные расширения для номенклатуры допускаются, только если стандартная номенклатура не включает специфические условия, требуемые приложением	Да/Нет (Если «Да»: объясните в подходящем ICS, см. 9.5.6)	
ADD-4	Формат полезной нагрузки	—	Были ли введены дополнительные форматы полезной нагрузки кроме тех, которые указаны в стандарте ИИЭР 11073-20601 или одной из перечисленных специализаций устройства?	Да/Нет (Если «Да», то объяснить полностью с указанием цели и компоновки. Объяснение должно быть выполнено на языке ASN.1)	

9.5.2 ICS объекта и класса (POC) информационной модели предметной области (DIM) персонального медицинского прибора

ICS POC определяет, какие из управляемых объектов из настоящего стандарта инстанцированы при реализации, и обозначает класс каждого объекта. Таблица 27 является шаблоном. Для каждого объекта, поддерживаемого реализацией, должен быть заполнен один ряд.

Таблица 26 — Шаблон для ICS POC

Индекс	Свойство	Ссылка	Статус	Поддержка	Комментарий
POC-*n*	Описание объекта	Класс объекта (такой как числовой класс и т.д.)	Реализованный	Обозначить ограничения (такие как максимальное число поддерживаемых экземпляров класса)	

Буква *n* в столбце «Индекс» должна обозначать дескриптор (handle) объекта для реализаций с предопределенными объектами. В иных случаях должна быть просто индивидуальным числом (1..*m*).

Все частные объекты должны быть обозначены и должны включать в себя ссылку на определение для объекта. Там, где нет возможности указать ссылки в открытом доступе, определение объекта должно быть приложено к заявлению о соответствии.

В столбце «Поддержка» должны быть указаны все ограничения для реализации объекта.

В качестве составной части ICS POC должно быть предоставлено графическое представление отношения включения объекта (графическое представление экземпляров класса).

9.5.3 ICS атрибутов POC

Для каждого поддерживаемого объекта, указанного в ICS POC, предоставляется ICS атрибутов POC для определения условных, необязательных или расширенных атрибутов, используемых/поддерживаемых реализацией, включая все наследуемые атрибуты. Обязательные атрибуты не нужно указывать, так как их соответствие требуется при реализации.

Таблица 27 является шаблоном.

Таблица 27 — Шаблон для ICS атрибутов POC

Индекс	Свойство	Ссылка	Статус	Поддержка	Комментарий
ATTR-*n*-*x*	Название атрибута. Расширенные атрибуты должны также включать в себя идентификаторы атрибута	Если атрибут не указан в настоящем стандарте или одной из перечисленных специализаций устройства, заполните ссылку к ASN.1	Реализованный	Описать: Доступ. Диапазон значений. Значения дополнительных ограничений	

Буква *n* в столбце «Индекс» — это идентификатор управляемого объекта, для которого составлена таблица (например, индекс управляемого объекта, как обозначено в ICS POC в 9.5.2). Для каждого поддерживаемого управляемого объекта составляется отдельная таблица.

Буква *x* в столбце «Индекс» — это серийный номер (1..*m*).

Все атрибуты, не входящие в настоящий стандарт или одну из перечисленных специализаций устройства, должны быть указаны и включать в себя ссылку на определение для атрибута. Там, где нет возможности указать ссылки в открытом доступе, определение атрибута должно быть приложено к декларации о соответствии. Области доступа спецификации атрибута в столбце «Поддержка» обозначены, если реализация предполагает сервисы доступа для атрибутов.

Столбец «Поддержка» должен также содержать диапазон значений атрибутов (в случае применимости), подсказки о специфических ограничениях на доступ атрибута или доступность атрибута и информацию, и обозначение того, является ли значение атрибута постоянной или переменной величиной при реализации.

Примечание — Таблицы с определениями атрибутов в настоящем стандарте определяют минимальный обязательный набор атрибутов для каждого объекта.

9.5.4 ICS поведения POC

ICS поведения POC определяет все методы реализованных объектов, которые можно вызвать сервисом ДЕЙСТВИЯ. Таблица 28 является шаблоном. Таблица составляется для каждого объекта, который поддерживает специальные методы.

Таблица 28 — Шаблон для ICS поведения POC

Индекс	Свойство	Ссылка	Статус	Поддержка	Комментарий
ACT-*n*-*x*	Название метода. Методы, не включенные в стандарты, должны также включать в себя Идентификатор метода	Если метод не указан в настоящем стандарте или одной из перечисленных специализаций устройства, заполните ссылку к ASN.1	Реализованный	Специфические ограничения	

Буква *n* в столбце «Индекс» — это идентификатор управляемого объекта, для которого составлена таблица (например, индекс управляемого объекта, как обозначено в POC ICS). Для каждого управ-

ляемого объекта, который поддерживает специфические методы объектов (то есть действий), составляется отдельная таблица.

Буква *x* в столбце «Индекс» — это серийный номер (1..*m*).

Все методы, не входящие в настоящий стандарт или одну из перечисленных специализаций устройства, должны быть указаны и включать в себя ссылку на определение для метода. Там, где нет возможности указать ссылки в открытом доступе, определение метода должно быть приложено к декларации о соответствии.

В столбце «Поддержка» должны быть обозначены все ограничения для метода.

9.5.5 ICS уведомлений POC

ICS уведомлений POC обозначает все реализованные уведомления (обычно в форме сервиса ОТЧЕТА О СОБЫТИИ), которые были вызваны поддерживаемыми объектами. Таблица 29 является шаблоном. Таблица составляется для каждого объекта, который поддерживает специальные уведомления для объекта.

Таблица 29 — Шаблон для ICS уведомления класса медицинского объекта (MOC)

Индекс	Свойство	Ссылка	Статус	Поддержка	Комментарий
NOTI-*n*-*x*	Название уведомления и идентификатор уведомления	Ссылка на подпункт настоящего стандарта, где указано событие		Специфические ограничения, идентификаторы и описание каждого вовлеченного объекта	

Буква *n* в столбце «Индекс» — это идентификатор управляемого объекта, для которого составлена таблица (например, индекс управляемого объекта, как обозначено в POC ICS). Для каждого управляемого объекта, который поддерживает специфические уведомления объектов (то есть событий), составляется отдельная таблица.

Буква *x* в столбце «Индекс» — это серийный номер (1..*m*).

Все частные уведомления должны быть указаны и включать в себя ссылку на определение для уведомления. Там, где нет возможности указать ссылки в открытом доступе, определение уведомления должно быть приложено к декларации о соответствии.

В столбце «Поддержка» должны быть обозначены все ограничения для уведомлений.

9.5.6 ICS номенклатур POC

ICS номенклатур POC указывает все реализованные номенклатуры, использованные агентом. Таблица 30 является шаблоном. Для каждого номенклатурного элемента должен быть использован отдельный ряд таблицы.

Таблица 30 — Шаблон для ICS номенклатур MOC

Индекс	Свойство	Ссылка	Статус	Поддержка	Комментарий
NOTI-*n*-*x*	Название уведомления и идентификатор уведомления	Ссылка на подпункт настоящего стандарта, где указано событие		Специфические ограничения, идентификаторы и описание каждого вовлеченного объекта	

Буква *n* в столбце «Индекс» — это последовательный номер для обеспечения уникальности (1..*m*).

Приложение А
(обязательное)

Определения языка ASN.1

А.1 Общие положения

Настоящее приложение предоставляет определения языка ASN.1, значимые для протоколов персональных медицинских приборов. Некоторые взяты из других частей серии стандартов ИСО/ИИЭР 11073, а другие созданы специально для области персональных медицинских приборов. Если необходимо понять, какие структуры импортированы, а какие являются новыми, см. приложение J. Настоящее приложение гарантирует, что все структуры данных, необходимые для реализации настоящего стандарта, легко доступны.

Соглашение о названиях, использованных в данном приложении, предполагает использование дефисов (-) для разделения слов в атрибутах и использование смешанного типа при указании типов данных; однако конструкции, взятые из других спецификаций, следуют существующим правилам использования заглавных букв и дефисов.

А.2 Общие типы данных

Данный подпункт определяет набор типов данных языка ASN.1, используемых в определениях объектов.

А.2.1 Целочисленные и битовые строковые типы данных

Для представления целых чисел, определения объектов используют только типы данных фиксированного размера. Битовые строковые типы данных представляют собой битовое поле, где каждый отдельный бит имеет определенное значение (например, поля метки). Используются следующие целочисленные типы данных и битовые строковые типы данных:

```
-- 8-битное беззнаковое целое число
--
INT-U8 ::= INTEGER (0..255)
--
-- 8-битное целое число со знаком
--
INT-I8 ::= INTEGER (-128..127)
--
-- 16-битное беззнаковое целое число
--
INT-U16 ::= INTEGER (0..65535)
--
-- 16-битное целое число со знаком
--
INT-I16 ::= INTEGER (-32768..32767)
--
-- 32-битное беззнаковое целое число
--
INT-U32 ::= INTEGER (0..4294967295)
--
-- 32-битное целое число со знаком
--
INT-I32 ::= INTEGER (-2147483648..2147483647)
--
-- 16-битная битовая строка
--
BITS-16 ::= BIT STRING (SIZE(16))
--
-- 32-битная битовая строка
--
BITS-32 ::= BIT STRING (SIZE(32))
```

Необходимо отметить, что в определениях объектов, целочисленных и битовых строковых типов данных с именованными константами или именованными битами для простоты используют обозначенные выше основные типы данных. Данный подход обеспечивает сокращенное обозначение, но такой синтаксис не допустим для языка ASN.1. Его можно легко преобразовать в правильный синтаксис. Например, определение

```
NamedConstant ::= INT-U16 {
                      const1(1),
              const2(2)
}
```

становится допустимым для языка ASN.1 синтаксисом в виде:

```
NamedConstant ::= INTEGER {
                      const1(1),
              const2(2)
} (0..65535)
```

А.2.2 Тип данных идентификации

Все элементы (например, классы, объекты и типы измерений), для которых необходима уникальная идентификация, получают идентификатор объекта (OID). Ряд действующих идентификаторов OID для настоящего стандарта определен в стандарте ИСО/ИИЭР 11073-10101 [12]. Номенклатура состоит из набора сегментов, в котором каждый сегмент относится к определенному понятию и имеет собственные 16-битные коды. Другими словами, специфический код определяется числом его сегментов и OID в рамках данного сегмента или его использование контекстно зависимо. В случае контекстно зависимых кодов специфический сегмент, используемый кодом, вызывается в рамках настоящего стандарта.

16-битный тип данных идентификации определяется следующим образом:

```
--
-- тип OID, как определено в номенклатуре
-- (не путать с идентификатором OID языка ASN.1)
--
OID-Type ::= INT-U16 -- 16-битный тип целых чисел
```

Для кодов и идентификаторов, которые еще предстоит стандартизировать, или для специфичных для производителя кодов существует частный сегмент.

```
--
-- Частный OID
--
PrivateOid ::= INT-U16
```

А.2.3 Тип данных дескриптора

Тип данных дескриптора используется для эффективного и уникального на местном уровне обозначения всех экземпляров управляемого объекта (уникальный на местном уровне означает уникальный в пределах контекста одной MDS). Этот тип данных определяется следующим образом:

```
--
-- дескриптор
--
HANDLE ::= INT-U16
```

А.2.4 Тип данных номера экземпляра

Номер экземпляра используется для разграничения экземпляров классов или объектов одного типа или экземпляров объекта, которыми нельзя управлять напрямую (используемых, например, в качестве идентифицирующих атрибутов для объектов PM-блока).

```
--
-- Количество экземпляров
--
InstNumber ::= INT-U16
```

А.2.5 Тип данных типа идентификатора

Тип данных типа идентификатора используется для определения типа всех элементов (например, классов, объектов и типов измерения). Он схож с типом OID (пункт В.2.2), но включает в себя и сегмент номенклатуры, и код для обеспечения уникальной идентификации элемента. Он должен быть использован, когда контекст не подразумевается. Этот тип данных определяется следующим образом:

```
--
-- Type ID
--
TYPE ::= SEQUENCE {
        сегмент NomPartition,
        код типа OID
}
--
-- Существуют следующие номенклатурные сегменты:
--
NomPartition ::= INT-U16 {
        nom-part-unspec(0),
        nom-part-obj(1),        -- объектно-ориентированный сегмент
        nom-part-metric(2),     -- метрический [супервизорное
                                -- управление и получение данных
```

```
                                -- (SCADA)] сегмент
        nom-part-alert(3),      -- сегмент сигналов/событий
        nom-part-dim(4),        -- сегмент размеров
        nom-part-vattr(5),      -- сегмент виртуальных атрибутов для
                                -- объектов функционирования
        nom-part-pgrp(6),       -- сегмент идентификатора группы
                                -- параметров
        nom-part-sites(7),      -- измерение и локализации участков —
                                -- тела
        nom-part-infrastruct(8),. -- сегмент инфраструктурных
                                -- элементов
        nom-part-fef(9)         -- сегмент формата обмена файлами
        nom-part-ecg-extn(10),  -- сегмент расширений
                                -- электрокардиограммы
        nom-part-phd-dm(128),   -- управление заболеваниями
        nom-part-phd-hf(129),   -- здоровье и физическая форма
        nom-part-phd-ai(130),   -- независимое старение
        nom-part-ret-code(255), -- сегмент кодов возврата
        nom-part-ext-nom(256),  -- идентификаторы для других
                                -- номенклатур и словарей
        nom-part-priv(1024)     -- частный сегмент
}
```

А.2.6 Тип данных установления значения атрибута (AVA)

Тип данных атрибута AVA полностью определяет атрибут объекта по его идентификатору атрибута и его значению. Так как структура значения зависит от атрибута, тип определяется через ANY DEFINED BY. Этот тип данных поддерживает ряд сервисов, используемых для доступа к атрибутам объекта (например, GET и SET). Значения идентификаторов атрибута определены для каждого типа объекта в столбце «Идентификатор атрибута» в таблицах определения объектов (например, таблица 2, таблицы 5—9, таблицы 12—15 и таблица 17). Структура, использованная для определения атрибута, определена в столбце «Тип атрибута» тех же таблиц. Тип данных AVA определяется следующим образом:

```
--
AVA-Type ::= SEQUENCE {
        attribute-id OID-Type, -- Его необходимо брать из сегмента
                                                --nom-part-obj
        attribute-value ANY DEFINED BY attribute-id
}
```

А.2.7 Тип данных списка атрибутов

Часто требуется список пар «Идентификатор атрибута — значение атрибута». Тип данных списка атрибутов — это специальный тип данных, предусмотренный для данной ситуации и определяемый следующим образом:

```
--
AttributeList ::= SEQUENCE OF AVA-Type
```

А.2.8 Тип данных списка идентификаторов атрибутов

Часто используется список идентификаторов атрибутов. Тип данных списка идентификаторов атрибутов — это специальный тип, предусмотренный для удобства и определяемый следующим образом:

```
--
AttributeIdList ::= SEQUENCE OF OID-Type
```

А.2.9 Тип данных с плавающей запятой (FLOAT-тип)

Тип данных с плавающей запятой (плавающий тип) определен для представления числовых значений нецелочисленного типа. Тип данных плавающего типа определен как 32-битное значение с 24-битной мантиссой и 8-битной экспонентой. См. F.7 для полного определения этого типа данных. Этот тип данных определяется следующим образом:

```
--
-- 32-bit float type; the integer type is a placeholder only
--
FLOAT-Type ::= INT-U32
```

32 бита содержат 8-битную экспоненту со знаком с основанием 10, за которой следует 24-битное целое число со знаком (мантисса).

Специальные значения закреплены для выражения следующего:

- NaN (Не число) [экспонента 0, мантисса +(2**23 —1) → 0x007FFFFF];
- NRes (не при этом разрешении экрана) [экспонента 0, мантисса –(2**23) → 0x00800000];
- + INFINITY (бесконечность) [экспонент 0, мантисса +(2**23 —2) → 0x007FFFFE];
- — INFINITY [экспонента 0, мантисса –(2**23 –2) → 0x00800002];
- Предназначено для дальнейшего использования [экспонента 0, мантисса —(2**23—1) → 0x00800001] 0x00800001].

А.2.10 Тип данных короткого плавающего типа (SFLOAT-тип)

Тип данных короткого плавающего типа определен для представления числовых значений нецелочисленного типа ограниченного разрешения. SFLOAT-тип определен как 16-битное значение с 12-битной мантиссой и 4-битной экспонентой. См. F.7 для полного определения этого типа данных. Этот тип данных определяется следующим образом:

```
--
-- 16-битный плавающий тип; целочисленный тип является только
--меткой-заполнителем
--
SFLOAT-Type ::= INT-U16
```

16-битное значение содержит 4-битную экспоненту с основанием 10, за которым следует 12-битная мантисса. Каждый из них выражен в форме дополнительного кода.

Специальные значения закреплены для выражения следующего:

- NaN [экспонента 0, мантисса +(2**11 –1) → 0x07FF];
- NRes [экспонента 0, мантисса –(2**11) → 0x0800];
- + INFINITY [экспонента 0, мантисса +(2**11 –2) → 0x07FE];
- — INFINITY [экспонента 0, мантисса –(2**11 —2) → 0x0802];
- Предназначено для дальнейшего использования [экспонента 0, мантисса –(2**11 –1) → 0x0801].

А.2.11 Тип данных относительного времени

Тип данных относительного времени это счетчик времени, используемый для определения относительного времени между событиями. Этот тип данных используется для расположения событий относительно друг друга. Он определяется следующим образом:

```
--
-- Относительное время имеет точность до 125 мкс (LSB), которой
-- достаточно для считывания показаний с частотой до 8 кГц,
-- охватывает периоды времени до 6.2 дней.
-- Должно быть использовано значение 0xFFFFFFFF, когда для отправки -- относительного времени на языке ASN.1 требуется агент,
-- но не поддерживает часы относительного времени.
--
RelativeTime ::= INT-U32
```

Следует учитывать, что точность реального времени определяется агентом.

А.2.12 Тип данных относительного времени высокой точности

Тип данных относительного времени высокой точности — это счетчик времени высокой точности, используемый для определения относительного времени между событиями. Этот тип данных используется для расположения событий относительно друг друга. Он определяется следующим образом:

```
--
-- Время высокой точности имеет точность до 1 мкс и может охватывать
-- периоды времени более 584 000 лет. Теоретически это может быть
-- смоделировано как INT-U64, однако из-за ограничения в
-- компиляторах языка ASN.1 встроенные устройства поддерживают
-- 64-битные целые числа и спецификации MDER, вместо этого была
-- использована строка октетов.
--
HighResRelativeTime ::= OCTET STRING (SIZE(8))
```

Необходимо отметить, что точность используемого реального времени определяется агентом.

```
--
-- Настройка абсолютного времени имеет точность до 1/100 секунды и
-- может представить настройки времени вперед или назад на 44 505
-- лет. Теоретически, это может быть смоделировано как INT-I48, однако
-- из-за возможного ограничения в компиляторах языка ASN.1
-- встроенные устройства поддерживают 48-битные целые числа и
-- спецификации MDER, вместо этого была использована строка октетов.
--
AbsoluteTimeAdjust ::= OCTET STRING (SIZE(6))
```

А.2.13 Тип данных абсолютного времени

Тип данных абсолютного времени определяет время дня с точностью до 1/100 секунды. Поле часов должно быть представлено в 24-часовом формате времени (т.е. от 0 до 23). Значения в структуре должны быть закодированы с использованием двоично-десятичного кодирования (т.е. 4-битных полубайтов). Например, 1996 год должен быть представлен в шестнадцатеричном значении 0x19 в поле века и шестнадцатеричном значении 0x96 в поле года. Этот формат легко конвертируется в формат, ориентированный на символы или целые числа. Тип данных абсолютного времени определяется следующим образом:

```
--
AbsoluteTime ::= SEQUENCE {
        век                 INT-U8,
        год                 INT-U8,
        месяц           INT-U8,
        день                INT-U8,
        час                 INT-U8,
        минута          INT-U8,
        секунда         INT-U8,
        доли секунды        INT-U8      -- 1/100 секунды, если доступно
}
```

Необходимо отметить, что точность реального времени определяется агентом (т. е. если точность часов 1 сек, то точность долей секунды всегда нулевая). Агенты должны иметь точность в 1 секунду или больше.

А.2.14 Тип данных рабочего состояния

Тип данных рабочего состояния определяет, включен или выключен определенный объект или какое-то другое свойство.

```
--
OperationalState ::= INT-U16 {
            disabled(0),
            enabled(1),
            notAvailable(2)
}
```

А.3 Тип данных атрибутов

А.3.1 Атрибуты системы MDS

```
--
-- SystemModel содержит имя производителя и информацию о модели,
-- специфичную для производителя. Несмотря на то, что поле номера
-- модели предполагает число, нет требования касательно того, что
-- поле должно содержать численное значение. Формат имени агента, но
-- он производителя и строки номера модели определяются поставщиком
-- должен быть печатаемым на ASCII.
--
SystemModel ::= SEQUENCE {
        manufacturer OCTET STRING,      -- размер строк должен быть четным
        model-number OCTET STRING       -- размер строк должен быть четным
}
--
-- ProductionSpec работает с серийными номерами, номерами деталей,
-- выпусками и т.д. Следует отметить, что агент может состоять из
-- нескольких компонентов; поэтому prod-spec должна быть строкой,
-- печатаемой на ASCII формата “spec-type: vendor-specified-str”, где
-- spec-type заменен строчным представлением spec-type. Формат
-- vendor-specified-str определяется поставщиком.
--
ProductionSpec ::= SEQUENCE OF ProdSpecEntry

ProdSpecEntry ::= SEQUENCE {
            spec-type           INT-U16 {
                            unspecified(0),
                            serial-number(1),
                            part-number(2),
                            hw-revision(3),
                            sw-revision(4),
                            fw-revision(5),
                            protocol-revision(6),
                            prod-spec-gmdn(7)   -- см. примечание по
                                                -- GMDN ниже
                            },
            component-id    PrivateOid,
            prod-spec       OCTET STRING        -- размер строк должен
                                                -- быть четным
}
```

```
-- Примечание — Всемирная номенклатура медицинских изделий (GMDN)
-- основана на ISO 15225 [15]и была разработана под покровительством
-- CEN TC257 SC1.4)
--
-- PowerStatus определяет, работает ли устройство на батарейках или
-- от сети. Старшие биты определяют состояние заряда.
--
PowerStatus ::= BITS-16 {
                onMains(0),
                onBattery(1),
                chargingFull(8),
                chargingTrickle(9),
                chargingOff(10)
}
--
-- Все измерения касательно батарейки являются значениями с их
-- размерами. См. описание Remaining-Battery-Time в таблице 2 для
-- описания допустимых единиц.
--
BatMeasure ::= SEQUENCE {
        value           FLOAT-Type,
        unit            OID-Type        -- из сегмента nom-part-dim
}
```

А.3.2 Метрические атрибуты

Данная группа содержит импортированные определения атрибутов, которые применяются к числовым, перечисляющим объектам и объектам RT-SA.

```
--
-- Статус измерения
-- Значения битов 14 и 15 используются в других стандартах ИСО/ИИЭР
-- 11073 и не должны использоваться для других целей.
--
MeasurementStatus ::= BITS-16 {
        invalid(0),
        questionable(1),
        not-available(2),
        calibration-ongoing(3),
        test-data(4),
        demo-data(5),
        validated-data(8), -- соответствующий, напр., в архиве
        early-indication(9), -- ранняя оценка значения
        msmt-ongoing(10) -- обозначает, что новые измерения еще
                         -- проводятся (случайный)
}
```

А.3.3 Числовые атрибуты

```
--
-- NuObsValue (Числовое наблюдаемое значение) всегда включает в себя
-- идентификацию, состояние и размер.
--
NuObsValue ::= SEQUENCE {
        metric-id       OID-Type,       -- Данный код взят из сегмента,
                                        -- обозначенного в атрибуте
                                        -- Metric::Type числового
                                        -- объекта.
        state           MeasurementStatus,
        unit-code       OID-Type,       -- из сегмента номенклатуры
                                        -- размеров nom-part-dim
        value           FLOAT-Type
}
--
-- Наблюдаемое значение для составных чисел
--
NuObsValueCmp ::= SEQUENCE OF NuObsValue
```

4) Дополнительную информацию о данном техническом комитете можно найти по адресу: gost.e-ecolog.ru

А.3.4 Атрибуты RT-SA

```
--
-- SaSpec описывает массив считываний
--
SaSpec ::= SEQUENCE {
        array-size              INT-U16,        -- количество считываний на каждый
                                                -- метрический период обновления
        sample-type             SampleType,
        flags                   SaFlags
}
--
-- SampleType описывает одно считывание в наблюдаемом массиве
--
SampleType ::= SEQUENCE {
            sample-size         INT-U8,         -- например, 8 для 8-битных
                                                -- считываний, 16 для
                                                -- 16-битных должны быть
                                                -- делимыми и на 8
            significant-bits    INT-U8          -- определяет значимые
                                                -- биты в одном считывании
                    { signed-samples(255)}      -- если значение 255,
                                                -- считывания в Simple-Sa-Observed-Value
                                                -- и lower-scaled-value и upper-scaled-value
                                                -- в ScaleRangeSpec должны восприниматься
                                                -- как целые числа со знаком в форме
                                                -- дополнительного кода.
}
--
-- SaFlags определяет дополнительные свойства форм сигнала.
--
SaFlags ::= BITS-16 {
        smooth-curve(0),                        -- для оптимального отображения исп.
                                                -- алгоритм кеширования
            delayed-curve(1),                   -- характеристическая кривая отложена (не-
                                                -- реальное время)
        static-scale(2),                        -- ScaleRangeSpec не меняется
        sa-ext-val-range(3)                     -- Незначимые биты в считывании не равны 0,
                                                -- напр., когда используются для аннотаций или
                                                -- маркеров. Получатель должен применить
                                                -- битовую маску для извлечения значимых
                                                -- битов из считывания. Если считывания
                                                -- имеют знак, бит sa-ext-val-range не
                                                -- должен быть установлен (т. к. в поле по
                                                -- определению не может быть незначительных
                                                -- битов).
}
--
-- Атрибут определения масштаба и диапазона описывает преобразование
-- между масштабируемыми и абсолютными значениями и определяет
-- ожидаемый диапазон абсолютных и масштабируемых значений.
-- В зависимости от диапазона масштабируемых значений, существует
-- множество типов атрибутов. Преобразование значений показаний
-- в конвертированные абсолютные значения определяется по формуле
-- Scale-and-Range-Specification в 6.3.5.3.
--
ScaleRangeSpec8 ::= SEQUENCE {
            lower-absolute-value        FLOAT-Type,
            upper-absolute-value        FLOAT-Type,
            lower-scaled-value          INT-U8,  -- n.b. интерпретируется
                                                 -- как INT-I8
            upper-scaled-value          INT-U8   -- если атрибут Sa-
                                                 -- Specification обозначает -
                                                 -- показания со знаком
}
```

```
ScaleRangeSpec16 ::= SEQUENCE {
        lower-absolute-value    FLOAT-Type,
        upper-absolute-value    FLOAT-Type,
        lower-scaled-value      INT-U16,        -- n.b. интерпретируется
                                                -- как INT-I16
upper-scaled-value              INT-U16         -- если атрибут Sa-
                                                -- Specification обозначает -
                                                -- показания со знаком
}
}
ScaleRangeSpec32 ::= SEQUENCE {
        lower-absolute-value    FLOAT-Type,
        upper-absolute-value    FLOAT-Type,
        lower-scaled-value      INT-U32,        -- n.b. интерпретируется
                                                --как INT-I32
upper-scaled-value              INT-U32         -- если атрибут Sa-
                                                -- Specification обозначает
                                                -- показания со знаком
}
```

А.3.5 Атрибуты перечисления

```
--
-- EnumObsValue описывает наблюдаемое значение перечисления.
--
EnumObsValue ::= SEQUENCE {
        metric-id       OID-Type,       -- Данный код взят из сегмента,
                                        -- обозначенного в атрибуте Metric-Id
                                        -- Partition в случае значимости.
                                        -- Если нет, то из того же сегмента, что и
                                        -- тип атрибута.
        state                            MeasurementStatus,
        value                            EnumVal        -- поддерживает различные типы
                                        --данных значений
}
-- EnumVal используется для обозначения различных конкретных
-- наблюдаемых типов данных следующим образом (необходимо отметить,
-- что тип измерения закодирован в структуре верхнего уровня
-- EnumObsVal::metric-id):
--
--   enum-obj-id:    используется для сообщения метрического OID,
-- например, в кач-ве аннотации или другого события, обозначенного в
-- сегменте Metric::Type;
--    enum-text-string: используется для сообщения строки
-- произвольного текста (например, сообщение о состоянии);
--    enum-bit-str:    для кодирования значений битовых строк; тип
-- данных битовых строк долженбыть определен отдельно, например, в
-- номенклатуре или в стандарте, специфичном для устройства.
--
-- Другие типы данных, обозначенные в ИСО/ИИЭР 11073-10201:2004 [13],
-- не включены сюда, т. к. они не являются значимыми для персональных
-- медицинских приборов.
--
EnumVal ::= CHOICE {
        enum-obj-id     [1] OID-Type,           -- Данный код взят из
        -- сегмента, обозначенного в атрибуте Enum-Observed-
        -- Value-Partition, если является значимым. Если нет, то из
        -- того же сегмента, что и атрибут Type.
        enum-text-string [2] OCTET STRING, -    - текст, печатаемый на
                                                -- ASCII, размер четный
        enum-bit-str    [16] BITS-32 — строка битов
}
```

А.3.6 Атрибуты сканера

Нет

А.3.7 Конфигурируемые атрибуты сканера

```
--
-- ConfirmMode определяет, используются ли отчеты подтвержденного
-- или неподтвержденного действия.
--
ConfirmMode ::= INT-U16 {
                unconfirmed(0),
                confirmed(1)
}
```

А.3.8 Случайные конфигурируемые атрибуты сканера

Нет

А.3.9 Повторяющиеся конфигурируемые атрибуты сканера

Нет

А.3.10 Атрибуты PM-блока и PM-сегмента

```
--
-- StoSampleAlg описывает то, как осуществляются считывания и их
-- усреднение.
--
StoSampleAlg ::= INT-U16 {
        st-alg-nos(0),                  -- не указано иное
        st-alg-moving-average(1),
        st-alg-recursive(2),
        st-alg-min-pick(3),
        st-alg-max-pick(4),
        st-alg-median(5),
        st-alg-trended(512),            -- используются значения направления
                                        -- развития
        st-alg-no-downsampling(1024), -- не предполагает усреднение,
                                        -- это реально измеренное показание
        st-alg-manuf-specific-start(61440), -- начало
                        -- зарезервированного диапазона, специфичного для
                        -- производителя
        st-alg-manuf-specific-end(65535) -- конец зарезервированного
                        -- диапазона, специфичного для производителя
}
```

А.4 Типы данных, связанных с методом ДЕЙСТВИЯ

```
--
-- SetTimeInvoke выбирает дату и время для установки.
--
SetTimeInvoke ::= SEQUENCE {
                date-time               AbsoluteTime,
                accuracy                FLOAT-Type -- приходится на установленное
                                        -- время (например, ошибка 2 мин); значение
                                        -- определяется в секундах. Данный параметр
                                        -- взят из ИСО/ИИЭР 11073-10201:2004 [13], но
                                        -- не используется. Таким образом, он будет равен нулю (0).
}
--
-- SegmSelection выбирает PM-сегменты, на которые распространяется
-- метод.
--
SegmSelection ::= CHOICE {
            all-segments    [1] INT-U16,        -- данный тип выбирается для
                -- выбора всех сегментов, действительное содержание поля
                -- — "не имеет значения" и должно быть равно нулю
            segm-id-list    [2] SegmIdList,     -- использование этого
                -- списка требует, чтобы менеджер уже знал атрибуты
                -- Instance-Number PM-сегментов, например, из предыдущего
                -- вызова метода Get-Segment-Info.
            abs-time-range [3] AbsTimeRange -- поддержка abs-time-range
                -- необязательна, обозначена в атрибуте PM-Store-Capab
}
```

```
--
-- SegmIdList выбирает PM-сегменты по идентификатору.
--
SegmIdList ::= SEQUENCE OF InstNumber
--
-- AbsTimeRange позволяет отбирать PM-сегменты по временному периоду.
--
AbsTimeRange ::= SEQUENCE {
                from-time       AbsoluteTime,
                to-time         AbsoluteTime
}
--
-- SegmentInfoList возвращает атрибуты объекта (за исключением
-- Fixed-Segment-Data) во всех выбранных экземплярах объекта PM
-- сегмента в ответ на метод PM-блока Get-Segment-Info.
-- Этого требует менеджер для нахождения динамичной информации о
-- сегментах.
--
SegmentInfoList ::= SEQUENCE OF SegmentInfo
SegmentInfo ::= SEQUENCE {
                seg-inst-no         InstNumber,
                seg-info        AttributeList
}
```

А.5 Типы данных, связанных с сообщением

```
ObservationScan ::= SEQUENCE {
                obj-handle  HANDLE,
                attributes  AttributeList
}
```

А.6 Прочие типы данных

```
--
-- TimeProtocolId обозначает протоколы времени,
-- поддерживаемые/используемые устройством.
--
TimeProtocolId ::= OID-Type -- из номенклатурного сегмента
-- nom-part-infrastruct
```

А.7 Кадр протокола персонального медицинского прибора

Следующий тип данных представляет собой кадр сообщения верхнего уровня протокола персонального медицинского прибора. Данные APDU (формирующиеся посредством PrstApdu) интерпретируются в соответствии с настоящим стандартом в результате переговоров, содержащихся в процедуре ассоциации, как описано в 8.7.3.1.

Правила кодирования MDER должны всегда распространяться на структуру в А.7.

```
ApduType ::= CHOICE {
        aarq        [57856]  AarqApdu,      -- Запрос на ассоциацию [0xE200]
        aare        [58112]  AareApdu,      -- Ответ на запрос на ассоциацию
                                            -- [0xE300]
        rlrq        [58368]  RlrqApdu,      -- Запрос на завершение ассоциации
                                            -- [0xE400]
        rlre        [58624]  RlreApdu,      -- Ответ на запрос на завершение
                                            -- ассоциации [0xE500]
        abrt        [58880]  AbrtApdu,      -- Принудит. прерывание ассоциации
                                            -- [0xE600]
        prst        [59136]  PrstApdu       -- Представление APDU [0xE700]
}
```

А.8 Определения протокола ассоциации

Правила кодирования MDER должны всегда распространяться на структуры в А.8.

```
AarqApdu ::= SEQUENCE {
                -- The assoc-version определяет версию ассоциации в
                -- процедуре ассоциации, использованной агентом. Агент
                -- должен установить точно одну версию бита. Если менеджер
                -- не поддерживает данную версию, он должен отклонить
```

```
        -- запрос на ассоциацию с rejected-unsupported-assoc-
        -- version.
        assoc-version           AssociationVersion,
        data-proto-list         DataProtoList
}
DataProtoList ::= SEQUENCE OF DataProto
-- Если data-proto-id установлен на data-proto-id-20601, data-proto-
-- info должен быть заполнен структурой PhdAssociationInformation.
-- Если data-proto-id установлен на data-proto-id-external, data-
-- proto-info должен быть заполнен структурой
-- ManufSpecAssociationInformation.
-- Если data-proto-id установлен на data-proto-id-empty, data-proto-
-- info должен быть пустым (используется, только если AareApdu
-- является отклонением).
DataProto ::= SEQUENCE {
        data-proto-id           DataProtoId,
        data-proto-info         ANY DEFINED BY data-proto-id
}
-- Все другие значения DataProtoId сохраняются и не должны быть
-- использованы.
DataProtoId ::= INT-U16 {
        data-proto-id-empty(0),   -- должен использоваться в
        -- AareApdu, только когда результат — отклонение
        data-proto-id-20601(20601),       -- обозначает, что протокол
        -- обмена следует настоящему стандарту
        data-proto-id-external(65535)  -- обозначает, что
        -- специфичный для производителя протокол данных УУИД
        -- является частью ManufSpecAssociationInformation
}
-- Ответ ассоциации
AareApdu ::= SEQUENCE {
        result                  AssociateResult,
        selected-data-proto     DataProto
}
-- Запрос выпуска
RlrqApdu ::= SEQUENCE {
        reason                  ReleaseRequestReason
}
-- Ответ выпуска
RlreApdu ::= SEQUENCE {
        reason                  ReleaseResponseReason
}
-- Прерывание
AbrtApdu ::= SEQUENCE {
        reason                  Abort-reason
}
-- Причина принудительного прерывания. Все значения «Abort-reason»
-- без знака предназначены для дальнейшего расширения и не должны
-- использоваться.
Abort-reason ::= INT-U16 {
            undefined(0),
            buffer-overflow(1),
            response-timeout(2),
            configuration-timeout(3)      -- Сообщение о конфигурации не
                                          -- было получено своевременно
}
-- См. 8.7.3.2 для описания использования. Все значения
-- «AssociateResult» без знака предназначены для дальнейшего
-- расширения и не должны использоваться.
AssociateResult ::= INT-U16 {
            accepted(0),
            rejected-permanent(1),
```

```
                rejected-transient(2),
                accepted-unknown-config(3),
                rejected-no-common-protocol(4),
                rejected-no-common-parameter(5),
                rejected-unknown(6),
                rejected-unauthorized(7),
                rejected-unsupported-assoc-version(8)
}
-- Все значения «ReleaseRequestReason» без знака предназначены для
-- будущего расширения и не должны использоваться.
ReleaseRequestReason ::= INT-U16 {
                normal(0),          -- используется, когда агент или менеджер
                                    -- решает завершить ассоциацию при
                                    -- нормальных условиях.
        no-more-configurations(1),  -- используется агентом, когда все
                                    -- возможные конфигурации были испробованы
                                    -- и менеджер отклонил их.
         configuration-changed(2)   -- используется агентом, когда
                                    -- изменения в его конфигурации требуют у
                                    -- агента завершения ассоциации. За этим
                                    -- может следовать Запрос на ассоциацию с
                                    -- информацией о новой конфигурации.
}
-- Все значения «ReleaseResponseReason» без знака предназначены для
-- дальнейшего расширения и не должны использоваться.
ReleaseResponseReason ::= INT-U16 {
            normal(0)
}
-- Значения Запроса на ассоциацию DataProto отображены на
-- PhdAssociationInformation. Эта информация используется для
-- заявления и обсуждения версии протокола, профиля и т.д.
PhdAssociationInformation ::= SEQUENCE {
    -- Информация protocolVersion используется для сообщения
    -- допустимых версий. Когда агент отправляет protocolVersion, он
    -- должен установить бит(ы) для каждой версии, которую он
    -- поддерживает. Когда менеджер отвечает, он должен установить один
    -- бит для обозначения версии, которая должна использоваться
    -- обоими. Если общей версии протокола нет, агент должен отклонить
    -- запрос ассоциации и установить protocolVersion на нули.
            protocol-version                ProtocolVersion,
            encoding-rules                  EncodingRules,
            nomenclature-version            NomenclatureVersion,
            functional-units                FunctionalUnits,
            system-type                     SystemType,
            system-id                       OCTET STRING,
            dev-config-id                   ConfigId,
            data-req-mode-capab             DataReqModeCapab,
            option-list                     AttributeList
}
--
-- Информация об ассоциации, специфичной для производителя, для
-- проприетарного протокола данных
--
ManufSpecAssociationInformation ::= SEQUENCE {
            data-proto-id-ext       UuidIdent,
            data-proto-info-ext     ANY DEFINED BY data-proto-id-ext
}
-- Все значения неназначенного бита «AssociationVersion»
-- предназначены для дальнейшего расширения и должны быть
-- установлены на ноль.
AssociationVersion ::= BITS-32 {
        assoc-version1(0)    -- Этот бит должен быть установлен,
        -- если поддерживается версия 1 протокола ассоциации
```

```
}
-- Все значения неназначенного бита «ProtocolVersion» предназначены
-- для дальнейшего расширения и должны быть установлены на ноль.
ProtocolVersion ::= BITS-32 {
        protocol-version1(0) -- Этот бит должен быть установлен,
                -- если поддерживается версия 1 протокола обмена
}
--
-- Агент и менеджер всегда должны поддерживать правила MDER. Агент и
-- менеджер могут рассмотреть другие правила кодирования помимо
-- правил MDER.Все значения бита «EncodingRules» без знака предназн.
-- для дальнейшего расширения и должны быть установлены на ноль.
EncodingRules ::= BITS-16 {
    mder(0)  -- Данный бит должен быть установлен, если MDER
             -- поддерживается /выбран
    xer(1),  -- Данный бит должен быть установлен, если XER
             -- поддерживается /выбран
    per(2)   -- Данный бит должен быть установлен, если PER
             -- поддерживается /выбран
}
-- Все значения бита «NomenclatureVersion» без знака предназначены
-- для дальнейшего расширения и должны быть установлены на ноль.
NomenclatureVersion ::=  BITS-32 { -- значения ссылаются на
                         -- специфический номенклатурный стандарт
        nom-version1(0) -- Данный бит должен быть установлен, если
                         -- поддерживается версия 1
}
-- Все значения бита «FunctionalUnits» без знака предназначены для
-- дальнейшего расширения и должны быть установлены на ноль.
FunctionalUnits ::= BITS-32 {
        fun-units-unidirectional(0),        -- предназначен для
        -- использования в дальнейшем. Данный бит должен быть
        -- установлен, если агент является однонаправленным.
        fun-units-havetestcap(1),-- Данный бит обозначает, может и
        -- устройство войти в тестовую ассоциацию.
        fun-units-createtestassoc(2)        -- Данный бит используется для
        -- обозначения намерения формирования тестовой ассоциации.
}
-- Все значения бита «SystemType» без знака предназначены для
-- дальнейшего расширения и должны быть установлены на ноль.
SystemType ::= BITS-32 {
          sys-type-manager(0),
          sys-type-agent(8)
}
ConfigId ::= INT-U16 {
        manager-config-response(0),
        standard-config-start(1),
        standard-config-end(16383),
        extended-config-start(16384),
        extended-config-end(32767),
        reserved-start(32768),
        reserved-end(65535)
}
```

А.9 Определения протокола представления данных

Правила кодирования MDER должны всегда распространяться на структуры в А.9.

```
--
-- OCTET STRING содержит APDU данных, закодированный согласно
-- абстрактному синтаксису и синтаксису передачи, оговоренным во
-- время ассоциации. Когда data-proto-id обсуждается на возможность
-- быть data-proto-id-20601, OCTET STRING должна быть закодированной
-- версией DataApdu.
--
PrstApdu ::= OCTET STRING
```

А.10 Определения протокола данных

А.10.1 Общая информация

В ходе процедуры ассоциации, описанной в разделе 8.7.3.1, было установлено, что пакет данных протокола прикладного уровня DataApdu и соответствующие структуры раздела А.10 должны соблюдать правила кодирования. Агент и менеджер обязаны соблюдать правила кодирования медицинских приборов, но также вправе рассмотреть и другие правила.

А.10.2 Блок протокола данных

```
--
-- Тип примитива комбинированного удаленного доступа и Тип операции
-- В сообщениях вызова удаленного доступа (roiv-*) идентификатор
-- вызова является неопределенными позволяет отправителю сообщения -- определить соответствующие ответные сообщения (если они есть).
-- Отправитель roiv-* сообщения должен выбрать значение
-- идентификатора вызова, необходимое для отличия данного сообщения -- от любого другого не устаревшего roiv-* сообщения. Сообщения
-- считаются устаревшими при получении ответа(rors-*, roer, или
-- rorj) или при превышении значения тайм-аута для подтверждения
-- сообщения. При возврате ответного сообщения (rors-*, roer, или
-- rorj), идентификатор сообщения вызова должен быть скопирован в
-- идентификатор вызова ответа. Это позволит инициатору подобрать
-- ответы к ожидающим запросам. Поскольку идентификатор является
-- неопределенным, получатель не сможет вычислить идентификатор
-- вызова. В частности, он не сможет предположить, что
-- идентификатор уникален для любой последовательности чисел и
-- периода времени.
--
DataApdu ::= SEQUENCE {
   invoke-id                          InvokeIDType,
   message                            CHOICE {
   roiv-cmip-event-report             [256] EventReportArgumentSimple, -- [0x0100]
   roiv-cmip-confirmed-event-report   [257] EventReportArgumentSimple, -- [0x0101]
   roiv-cmip-get                      [259] GetArgumentSimple, -- [0x0103]
   roiv-cmip-set                      [260] SetArgumentSimple, -- [0x0104]
   roiv-cmip-confirmed-set            [261] SetArgumentSimple, -- [0x0105]
   roiv-cmip-action                   [262] ActionArgumentSimple, -- [0x0106]
   roiv-cmip-confirmed-action         [263] ActionArgumentSimple, -- [0x0107]
   rors-cmip-confirmed-event-report   [513] EventReportResultSimple, -- [0x0201]
   rors-cmip-get                      [515] GetResultSimple,-- [0x0203]
   rors-cmip-confirmed-set            [517] SetResultSimple,-- [0x0205]
   rors-cmip-confirmed-action         [519] ActionResultSimple, -- [0x0207]
   roer                               [768] ErrorResult,-- [0x0300]
   rorj                               [1024] RejectResult -- [0x0400]
   }
}
-- Отправитель должен ограничить количество сообщений, одновременно
-- ожидающих ответа. Фактически,принимающая сторона вероятнее всего
-- сможет обрабатывать не более одного сообщения за раз
InvokeIDType ::= INT-U16

-- Если в результате действия, вызванного DataApdu (roiv-*)
-- произошла ошибка, получатель отправляет ErrorResult.
-- Идентификатор вызова invokeID используется для определения
-- вызова, в котором произошла ошибка. Код ошибки указывается из
-- приведенного ниже списка RoerErrorValue. Параметры заполняются
-- дальнейшими данными, если это позволяет код ошибки. Использование
-- значения параметра указано в комментариях RoerErrorValue.
ErrorResult ::= SEQUENCE {
        error-value     RoerErrorValue,
        parameter               ANY DEFINED BY error-value
}
```

```
-- Все неназначенные значения «RoerErrorValue» сохраняются для
-- дальнейшего расширения и неиспользуются. Необходимо отметить,
-- что в стандарте ИСО/ИИЭР 11073-20101:2004 [14] указан ряд
-- значений RoerErrorValue, не упомянутых в настоящем стандарте. В
-- целях постоянства, в нумерации значений RoerErrorValue
-- пропускаются все значения уже указанные в ИСО/ИИЭР 11073-
-- 20101:2004.
RoerErrorValue ::= INT-U16 {
    -- значение no-such-object-instance возвращается, когда ссылается
    -- на недопустимое использование или на попытку доступа к любому
    -- объекту, кроме системы медицинского прибора, до утверждения
    -- конфигурации, т.е. агент и менеджер находятся не в рабочем
    -- состоянии.
    no-such-object-instance(1),
    -- значение no-such-action возвращается, когда команда действия
    -- является незаконной
    no-such-action(9),
    -- значение invalid-object-instance возвращается, когда объект
    -- существует, но команда является незаконной для данного типа
    -- объекта (например, Применяется к любому объекту кроме системы
    -- медицинского прибора или РМ-блока).
    invalid-object-instance (17),
    -- значение protocol-violation возвращается в случаях нарушения
    -- протокола (например, размер пакета данных протокола
    -- прикладного уровня превышает максимальное значение)
    protocol-violation(23),
    -- значение not-allowed-by-object возвращается при попытке
    -- применения действия к объекту, но объект не разрешает
    -- выполнение данного действия. На более высоком уровне возможно
    -- отображение причины прерывания действия в виде типа
    -- идентификатора объекта в поле параметра, в результате
    -- использования кода возврата, взятого из раздела кода
    -- завершения
    not-allowed-by-object(24),
    -- значение action-timed-out возвращается при прерывании
    -- действия до его завершения или если время выполнения действия
    -- превысит установленное значение тайм-аута. На более высоком
    -- уровне возможно отображение причины прерывания действия в виде
    -- типа идентификатора объекта в поле параметра, в результате
    -- использования кода возврата, взятого из раздела кода
    -- завершения
    action-timed-out(25),
    -- значение action-aborted возвращается, если действие прервано
    -- по причинам на более высоком уровне (например, превышен объем
    -- памяти). На более высоком уровне возможно отображение причины
    -- прерывания действия в виде типа идентификатора объекта в поле
    -- параметра, в результате использования кода возврата, взятого
    -- из раздела кода завершения
    action-aborted(26)
}
-- The RejectResult используется, если сообщение было отклонено
RejectResult ::= SEQUENCE {
problem RorjProblem
}
-- Все не назначенные значения «RorjProblem» сохраняются для
-- дальнейшего расширения и не используются.
RorjProblem ::= INT-U16 {
    -- значение unrecognized-apdu возвращается, если пакет данных
    -- протокола прикладного уровня DataApdu нераспознан
    unrecognized-apdu(0),
    -- значение badly-structured-apdu возвращается, если получатель
    -- не может распознать пакет данных протокола прикладного уровня
```

```
    -- DataApdu в связи с его структурой (или ее отсутствия)
    -- (например, некорректный размер данных)
    badly-structured-apdu(2),
    -- значение unrecognized-operation отправляется, если получатель
    -- не может распознать запрошенную операцию
    unrecognized-operation(101),
    -- значение resource-limitation отправляется, если получатель не
    -- может обработать сообщение в виду ограниченного ряда причин.
    resource-limitation(103),
    -- значение unexpected-error отображает ошибочное условие в
    -- случаях, если невозможно определить точный код ошибки
    unexpected-error(303)
}
```

А.10.3 Сервис УВЕДОМЛЕНИЙ О СОБЫТИЯХ

```
-- Идентификатор obj-handle для уведомлений, описанных в данном
-- стандарте, должно иметь значение равное 0 для отображения объекта
-- системы медицинского прибора либо отображать сканирующее
-- устройство или объект PM-блока. Если агент не поддерживает
-- RelativeTime (как указано в бите mds-time-capab-relative-time в
-- MdsTimeCapState), времени событий event-time должно быть
-- присвоено значение 0xFFFFFFFF. Менеджерам следует игнорировать
-- время событий event-time, если агент сообщает, что он не
-- поддерживает RelativeTime. Для различных типов событий event-
-- types, указанных в таблице 4, таблице 11, таблице 16 и таблице
-- 18, используется соответствующая структура данных о событии event-
-- info. Соответственно, значение event-info может быть одним из
-- следующих ConfigReport, ScanReportInfoFixed, ScanReportInfoVar,
-- ScanReportInfoMPFixed, ScanReportInfoMPVar,
-- ScanReportInfoGrouped, ScanReportInfoMPGrouped, или
-- SegmentDataEvent
EventReportArgumentSimple ::= SEQUENCE {
        obj-handle          HANDLE,
        event-time          RelativeTime,
        event-type          OID-Type,           -- Из раздела nom-part-obj
                                                -- Подраздел NOTI (MDC_NOTI_*)
        event-info          ANY DEFINED BY event-type
}

-- Идентификатор obj-handle для уведомлений, описанных в данном
-- стандарте, должен иметь значение равное 0 для отображения объекта
-- системы медицинского прибора либо отображать сканирующее
-- устройство или объект PM-блока.Тип события (event-type)
-- результатов должен быть копией типа события event-type от
-- вызова. Для различных типов событий event-types, указанных в
-- таблице 4, таблице 11, таблице 16 и таблице 18 используется
-- соответствующее значение event-reply-info. Соответственно,
-- значение event-reply-info не заполняется, либо должно быть
-- ConfigReportRsp, или SegmentDataResult.
EventReportResultSimple ::= SEQUENCE {
        obj-handle      HANDLE,
        currentTime     RelativeTime,
        event-type  OID-Type,           -- Из раздела nom-part-obj
                                        -- Подраздел NOTI (MDC_NOTI_*)
event-reply-info        ANY DEFINED BY event-type
}
```

А.10.4 Сервис GET

```
-- В отношении запросов GET, обозначенных в настоящем стандарте,
-- obj-handle должен иметь значение равное 0 для отображения объекта
-- системы медицинского прибора либо отображать объект PM-блока.
-- Параметр attribute-id-list не заполняется для запроса всех
-- атрибутов системы медицинского прибора или объекта PM-блока. В
-- ином случае, специфические атрибуты объекта могут быть запрошены
-- путем составлением списка желаемых идентификаторов атрибута
-- Attribute ID, которые указаны в таблице 2 или таблице 9.
```

```
GetArgumentSimple ::= SEQUENCE {
        obj-handle              HANDLE,
        attribute-id-list       AttributeIdList
}
-- Идентификатор obj-handle для запросов GET, описанных в данном
-- стандарте, должен соответствовать одному из запросов. Список
-- атрибутов attribute-list содержит все запрашиваемые запросы в
-- различных форматах. Если запрашиваемый идентификатор атрибута
-- отсутствует в объекте системы медицинского прибора, он не должен
-- возвращаться в список атрибутов attribute-list.
GetResultSimple ::= SEQUENCE {
        obj-handle              HANDLE,
        attribute-list          AttributeList
}
TypeVerList ::= SEQUENCE OF TypeVer
-- Поскольку тип должен быть выбран из ИСО/ИИЭР 11073-10101 [12],
-- раздела передачи nom-part-infrastruct, подраздела DEVspec,
-- предпочтительно использование обычного типа идентификатора
-- объекта -OID-Type, а не TYPE. В частных специализациях ИИЭР
-- 11073-104zz указано, какая специализация относится к
-- версии 1, 2, ..., и так далее; таким образом версия 3 может
-- соответствовать версии спецификации 1.5.
TypeVer ::= SEQUENCE {
        type            OID-Type,
        version         INT-U16
}
```

A.10.5 Сервис SET

```
-- Идентификатор obj-handle для сервисов GET, описанных в данном
-- стандарте, должен иметь значение, указывающее на сканирующее
-- устройство.
SetArgumentSimple ::= SEQUENCE {
        obj-handle              HANDLE,
        modification-list       ModificationList
}
ModificationList ::= SEQUENCE OF AttributeModEntry
AttributeModEntry ::= SEQUENCE {
        modify-operator         ModifyOperator,
        attribute               AVA-Type
}
-- Все не назначенные значения «ModifyOperator» сохраняются для
-- дальнейшего расширения и не используются.
ModifyOperator ::= INT-U16 {
         replace(0),
        addValues(1), -- используется для изменения значений,
        -- содержащихся в категориях данных спискового вида
        removeValues(2), -- используется для изменения значений,
        -- содержащихся в категориях данных спискового вида
        setToDefault(3)
}
--
-- Значение идентификатора obj-handle должно соответствовать
-- значению, полученному в SetArgumentSimple. Список атрибутов
-- attribute-list содержит все идентификаторы атрибутов, которые
-- были изменены и возвращает новое значение атрибута. Обычно это
-- значение берется из команды Set; однако, в виду округления или
-- ошибки, возвращаемое значение может отличаться от запрошенного.
SetResultSimple ::= SEQUENCE {
         obj-handle             HANDLE,
         attribute-list         AttributeList
}
```

A.10.6 Сервис ACTION

```
-- Идентификатор obj-handle для запросов действия, описанных в
-- настоящем стандарте, должен иметь значение равное 0 для
-- отображения объекта системы медицинского прибора либо отображать
-- объект PM-блока. Для типов действия action-types, указанных в
-- таблице 3 и таблице 10, используются соответствующие структуры
-- action-info-args. Соответственно, значение action-info-args
-- должно быть одним из DataRequest, SetTimeInvoke, SegmSelection,
-- или TrigSegmDataXferReq.
ActionArgumentSimple ::= SEQUENCE {
        obj-handle          HANDLE,
        action-type         OID-Type,       -- Из раздела nom-part-obj
                                            -- Subpartition ACT (MDC_ACT_*)
        action-info-args ANY DEFINED BY action-type
}

-- Идентификатор obj-handle для ответов действий, указанных в данном
-- стандарте, должен совпадать с идентификатором соответствующего
-- запроса.
-- Тип действия action-type должен копироваться из типа действия
-- сообщения вызова.
-- Для типов действия action-types, указанных в
-- таблице 3 и таблице 10 используется полученное значение action-
-- info-args. Соответственно, значение action-info-args не
-- заполняется, либо должно быть DataResponse, SegmentInfoList, или
-- TrigSegmDataXferRsp.
ActionResultSimple ::= SEQUENCE {
        obj-handle          HANDLE,
        action-type         OID-Type,       -- Из раздела nom-part-obj
                                            -- Subpartition ACT (MDC_ACT_*)
        action-info-args ANY DEFINED BY action-type
}
```

A.11 Типы данных атрибутов нового объекта и сервисов объектов

A.11.1 Основные типы данных

```
AttrValMap ::= SEQUENCE OF AttrValMapEntry
AttrValMapEntry ::= SEQUENCE {
        attribute-id OID-Type,      -- Получается из раздела nom-part-obj
        attribute-len INT-U16
}
```

A.11.2 Типы данных системы медицинского прибора

```
UuidIdent ::= OCTET STRING(SIZE(16))
-- Точность синхронизации времени time-sync-accuracy позволяет
-- агенту сообщать насколько точно синхронизированы его внутренние
-- часы в соответствии с настройками мастера синхронизации (если
-- используется синхронизация времени).
MdsTimeInfo ::= SEQUENCE {
        mds-time-cap-state      MdsTimeCapState,
        time-sync-protocol      TimeProtocolId, -- это номенклатурный
                                -- код из раздела nom-part-infrastruct
        time-sync-accuracy      RelativeTime, -- 0xFFFFFFFF если
                                -- неизвестно; 0 если лучше, чем 1/8 мс
        time-resolution-abs-time INT-U16, -- Разрешающая способность
                                -- часов абсолютного времени агента.
                                -- 0, если неизвестно; в противном
                                -- случае,1/100 сек, истекающие с каждым
                                -- делением часов. Например, если часы
                                -- агента тикают с интервалом равным
                                -- 1 с, то берется значение равное 100.
        time-resolution-rel-time INT-U16,   -- Разрешающая способность
                                -- часов относительного времени агента
                                -- 0, если неизвестно;
```

```
                                -- в противном случае, 125 µs истекающие
                                -- с каждым делением часов. Например,
                                -- если часы агента тикают с интервалом
                                -- равным 1, то берется значение равное
                                -- 8000.
        time-resolution-high-res-time INT-U32    -- Разрешающая
                                -- способность часов агента с высоким
                                -- разрешением времени 0, если
                                -- неизвестно; в противном случае,
                                -- количество микросекунд, проходящих
                                -- с каждым делением часов. Например,
                                -- если часы агента тикают с интервалом
                                -- равным 1, берется значение
                                -- равное 1 000 000.
}
-- Все не назначенные значения битов «MdsTimeCapState» сохраняются
-- для дальнейшего расширения и равны нулю.
MdsTimeCapState ::= BITS-16 {
        mds-time-capab-real-time-clock(0), -- устройство оснащено
                                -- внутренними часами реального времени
        mds-time-capab-set-clock(1),        -- устройство поддерживает
                                -- настройку времени Set Time Action
        mds-time-capab-relative-time(2),    -- устройство поддерживает
                                -- относительное время RelativeTime
        mds-time-capab-high-res-relative-time(3),     -- устройство
                                -- поддерживает HighResRelativeTime
        mds-time-capab-sync-abs-time(4),    -- устройство синхронизирует
                                -- AbsoluteTime
        mds-time-capab-sync-rel-time(5),    -- устройство синхронизирует
                                -- RelativeTime
        mds-time-capab-sync-hi-res-relative-time(6),-- устройство
                                -- синхронизирует HiResRelativeTime
        mds-time-state-abs-time-synced(8),  -- AbsoluteTime
                                -- синхронизировано
        mds-time-state-rel-time-synced(9),  -- RelativeTime
                                -- синхронизировано
        mds-time-state-hi-res-relative-time-synced(10),
                                -- HiResRelativeTime синхронизировано
        mds-time-mgr-set-time(11) -- менеджер стимулирован к настройке
                                -- времени
}
-- ************
-- Список различных элементов, связанных с соответствием нормативным
-- документам и сертификации, на которые ссылается агент.
-- ************
RegCertDataList ::= SEQUENCE OF RegCertData
RegCertData ::= SEQUENCE {
        auth-body-and-struc-type        AuthBodyAndStrucType,
        auth-body-data                  ANY DEFINED BY auth-body-and-struc-type
}

AuthBodyAndStrucType ::= SEQUENCE {
        auth-body                       AuthBody,
        auth-body-struc-type            AuthBodyStrucType
}

-- не определенные значения «AuthBody» сохраняются для дальнейшего
-- расширения и не используются.

AuthBody ::= INT-U8 {
        auth-body-empty(0),
        auth-body-ИИЭР -11073(1),
        auth-body-continua(2),
```

```
        auth-body-experimental(254),
        auth-body-reserved(255)
}
--
-- Другие возможные/предполагаемые авторитетные органы
-- auth-body-eu(),
-- auth-body-ИИЭР (),
-- auth-body-iso(),
-- auth-body-us-fda(),
-- когда установленный авторитетный орган назначает свой первый тип
-- AuthBodyStrucType для особого значения auth-body-data, ему
-- следует использовать специфические значения.

-- AuthBodyStrucType контролируется и назначается авторитетным
-- органом
AuthBodyStrucType ::= INT-U8
```

A.11.3 Типы данных, связанные с измерением

```
--
-- SupplementalTypeList предоставляет расширяемый механизм записи
-- дополнительной информации о объекте. Эта категория может
-- содержать такую информацию, как расположение сенсора или
-- чувствительность объекта.
--
SupplementalTypeList ::= SEQUENCE OF TYPE
--
-- В соответствии с ИСО/ИИЭР 11073-10201:2004 [13] атрибут Metric
-- Spec Small это сокращенный атрибут MetricSpec. Он отражает
-- доступность, периодичность и категорию измерения. Если метрика
-- соответствует значениям по умолчанию (не установлены двоичные
-- разряды), то этот атрибут не требуется.Все не назначенные
-- значения «MetricSpecSmall» сохраняются для дальнейшего расширения
-- и должны быть равны нулю.
--
MetricSpecSmall ::= BITS-16 {
    mss-avail-intermittent(0), -- значение доступно только
                               -- периодически
    mss-avail-stored-data(1), -- Агент может хранить и отправлять
                               -- несколько ранее фиксированных значений
                               -- (например, шкала весов хранит до 25
                               -- значений)
    mss-upd-aperiodic(2), -- значение устанавливается только
                               -- апериодически (например, только при
                               -- изменении)
    mss-msmt-aperiodic(3), -- апериодическое измерение
    mss-msmt-phys-ev-id(4), -- измерение является физиологическим
                               -- переключателем(например, для
                               -- обозначения определения пульса)
    mss-msmt-btb-metric(5), -- измерение от удара к удару или от
                               -- вдоха к выдоху
    mss-acc-manager-initiated (8), -- существует доступ к значению
                               -- объекта через передачу данных, запущенную
                               -- менеджером
    mss-acc-agent-initiated(9), -- значение объекта обновляется с
                               -- помощью передачи данных, запущенной
                               -- агентом
    -- П р и м е ч а н и е — касательно использования следующих битов mss-
    -- cat-* Биты mss-cat-setting imss-cat-calculation нельзя
    -- устанавливать для автоматического измерения. Метрика
    -- представляет собой нормально, регулярно измеряемое значение.
    -- Это значит, что ни один из битов mss-cat-*не
    -- устанавливается (по умолчанию) для автоматически получаемых
    -- измерений, проводимых агентом.
```

```
        mss-cat-manual(12),  -- Если установлен данный бит, то метрика
                             -- получается вручную (например, пользователь
                             -- вручную задает значение), если данный бит
                             -- неустановлен, то метрика получается
                             -- автоматически (например, устройство
                             -- измеряет значение).
        mss-cat-setting(13),  -- Если установлен этот бит, метрика
                             -- представляет собой параметры устройства.
                             -- Это может быть значение установленное
                             -- автоматически или вручную, как указано в
                             -- бите mss-cat-manual.
        mss-cat-calculation(14) -- Если установлен этот бит, метрика
                             -- представляет собой рассчитанное значение,
                             -- рассчитанное автоматически или вручную,
                             -- как указано в бите mss-cat-manual. Такие
                             -- значения рассчитываются из результатов
                             -- автоматического измерения и/или из
                             -- значений, введенных вручную.
}

-- Данный атрибут частично взят из ИСО/ИИЭР 11073-10201 [13] и
-- усовершенствован с помощью значения ms-struct::fix-comp-no.
--
MetricStructureSmall ::= SEQUENCE {
        ms-struct INT-U8 {
                ms-struct-simple(0),
                ms-struct-compound(1), -- неоднократно наблюдаемое
                                       -- значение,одинаковый динамический
                                       -- контекст
                ms-struct-reserved(2),  -- для ИСО/ИИЭР 11073-10201:2004
                ms-struct-compound-fix(3) -- близкий к составному (1), но
                                       -- размер массива наблюдаемого
                                       -- составного значения не должен быть
                                       -- динамичным в ходе ассоциации.
        },
        ms-comp-no INT-U8              -- максимальное количество
                                       -- компонентов/элементов в
                                       -- наблюдаемом составном значении, 0
                                       -- если ms-struct настроен на
                                       -- ms-struct-simple
                }
-- Данный атрибут определяет список идентификаторов метрики
-- MetricIds.
--
MetricIdList ::= SEQUENCE OF OID-Type

--
-- EnumPrintableString это тип данных для отображения Подсчета
-- Наблюдаемых Значений в форме текстовых фрагментов, пригодных для
-- печати, в формате ASCII.
--
EnumPrintableString ::= OCTET STRING      -- размер текстового
                                       -- фрагмента должен быть целым числом
PersonId ::= INT-U16 {
            unknown-person-id(65535)          -- 0xFFFF
}
```

А.11.4 Типы данных, связанные со сканером

```
HandleAttrValMap ::= SEQUENCE OF HandleAttrValMapEntry
HandleAttrValMapEntry ::= SEQUENCE {
             obj-handle        HANDLE,
             attr-val-map      AttrValMap
}
HANDLEList ::= SEQUENCE OF HANDLE
```

А.11.5 Сервисы системы медицинского прибора

```
-- Следующие определения поддерживают определения
-- EventReportArgumentSimple и ActionArgumentSimple.
--
-- Типы Информации о сканировании Scan Report Info используются в
-- качестве типов данных результатов от различной
-- последовательности событий MDS-Dynamic-Data-Update* (см.
-- раздел 6.3.2.5).
--

-- Определения ScanReport* используются при передаче информации о
-- выбранных измерениях. Существует два вектора: A) одна персона или
-- несколько и B) переменный формат, фиксированный формат или
-- группированный формат. Комбинация этих векторов приводит к шести
-- определениям высшего уровня: ScanReportInfoVar,
-- ScanReportInfoFixed, ScanReportInfoGrouped,
-- ScanReportInfoMPVar, ScanReportInfoMPFixed, и
-- ScanReportInfoMPGrouped.
-- ПОСЛЕДОВАТЕЛЬНОСТЬ ObservationScan или ObservationScanFixed может
-- содержать несколько экземпляров одного и того же идентификатора,
-- пока у них есть отличительные временные метки. Во всех случаях
-- значение scan-report-no должно быть аннулировано во время
-- ассоциации и постепенно должно повышаться на единицу до
-- одновременного нажатия нескольких клавиш.
----------------------------------------------------------------------------
ScanReportInfoVar ::= SEQUENCE {
        data-req-id             DataReqId,
        scan-report-no          INT-U16,        -- счетчик для отслеживания
                                                -- пропавших результатов сканирования
        obs-scan-var            SEQUENCE OF ObservationScan
}
----------------------------------------------------------------------------
ScanReportInfoFixed ::= SEQUENCE {
        data-req-id             DataReqId,
        scan-report-no          INT-U16,        -- счетчик для отслеживания
                                                -- пропавших результатов сканирования
        obs-scan-fixed          SEQUENCE OF ObservationScanFixed
}

ObservationScanFixed ::= SEQUENCE {
        obj-handle              HANDLE,         -- уникальная идентификация
                                                -- метрического объекта
        obs-val-data            OCTET STRING -- наблюдаемые данные значения,
                                                -- определяемые obj-handle
}
----------------------------------------------------------------------------
-- obs-scan-grouped это ПОСЛЕДОВАТЕЛЬНОСТЬ таких периодических
-- измерений, которые могут объединять несколько отчетов в одном.
-- Периодические отчеты должны быть составлены таким образом, чтобы
-- по возможности не требовалось помещать несколько отчетов в один
-- отчет о сканировании.
ScanReportInfoGrouped ::= SEQUENCE {
        data-req-id             INT-U16,
        scan-report-no          INT-U16,        -- счетчик для отслеживания
                                                -- пропавших результатов сканирования
        obs-scan-grouped        SEQUENCE OF ObservationScanGrouped
}

ObservationScanGrouped ::= OCTET STRING -- Формат определяется
                                        -- HandleAttrValMap
----------------------------------------------------------------------------
```

```
ScanReportInfoMPVar :: = SEQUENCE {
        data-req-id             DataReqId,
        scan-report-no          INT-U16,            -- счетчик для отслеживания
                                                    -- пропавших результатов сканирования
        scan-per-var            SEQUENCE OF ScanReportPerVar
}

DataReqId ::= INT-U16 {
        data-req-id-manager-initiated-min(0),               -- 0x0000
        data-req-id-manager-initiated-max(61439),           -- 0xEFFF
        -- Значения между data-req-id-manager-initiated-min и
        -- data-req-id-manager-initiated-max, включительно, должны
        -- использоваться в ходе передачи данных об измерениях,
        -- проводимой менеджером.
        --
        data-req-id-agent-initiated(61440)                  -- 0xF000
        -- data-req-id-agent-initiated должно использоваться
        -- в ходе передачи данных об измерениях, проводимой агентом.
        --
        -- Значения между 0xF001 и 0xFFFF, включительно,
        -- зарезервированы.
}

--
-- Значение, используемое для идентификации личности определяется
-- поставщиком (например, если у агента есть две кнопки для
-- идентификации двух лиц, то агент может использовать
-- идентификаторы 1 и 2 или идентификаторы 35 и 97). В настоящем
-- стандарте не описывается процесс присвоения идентификатора
-- отдельным лицам.
--
ScanReportPerVar ::= SEQUENCE {
        person-id       PersonId,
        obs-scan-var  SEQUENCE OF ObservationScan
}
-------------------------------------------------------------------------------
ScanReportInfoMPFixed ::= SEQUENCE {
        data-req-id             DataReqId,
        scan-report-no          INT-U16,            -- счетчик для отслеживания
                                                    -- пропавших результатов сканирования
        scan-per-fixed          SEQUENCE OF ScanReportPerFixed
}
ScanReportPerFixed ::= SEQUENCE {
        person-id               PersonId,
        obs-scan-fixed          SEQUENCE OF ObservationScanFixed
}
-------------------------------------------------------------------------------
ScanReportInfoMPGrouped ::= SEQUENCE {
        data-req-id             INT-U16,
        scan-report-no          INT-U16,            -- счетчик для отслеживания
                                                    -- пропавших результатов сканирования
        scan-per-grouped        SEQUENCE OF ScanReportPerGrouped
}
ScanReportPerGrouped ::= SEQUENCE {
        person-id               PersonId,
        obs-scan-grouped ObservationScanGrouped
}
-------------------------------------------------------------------------------
-- Определение ConfigReport используется при передаче данных о
-- настройках агента менеджеру. (см. таблицу 4)
ConfigReport ::= SEQUENCE {
       config-report-id         ConfigId,
       config-obj-list          ConfigObjectList
}
```

```
ConfigObjectList ::= SEQUENCE OF ConfigObject
ConfigObject ::= SEQUENCE {
        obj-class               OID-Type, -- Из раздела nom-part-obj
                                -- Подраздел MOC/BASE (MDC_MOC_VMD_*)
        obj-handle              HANDLE,
        attributes              AttributeList
}
ConfigReportRsp ::= SEQUENCE {
        config-report-id        ConfigId,
        config-result           ConfigResult
}
-- Не определенные значения «ConfigResult» сохраняются для
-- дальнейшего расширения и не используются.
ConfigResult ::= INT-U16 {
        accepted-config(0),
        unsupported-config(1),
        standard-config-unknown(2)
}
DataRequest ::= SEQUENCE {
        data-req-id             DataReqId, -- Позволяет отличать ответы от
                                        -- нескольких запросов данных (если
                                        -- устройство поддерживает несколько
                                        -- одновременных запросов данных).
                                        -- Отраженный в ScanReportInfo*
                                        -- идентификатор data-req-id.
        data-req-mode           DataReqMode, -- определяет режим установкой
                                        -- одного или нескольких битов.
        data-req-time           RelativeTime, -- Отображает сколько времени
                                        -- выделено агенту для передачи данных.
                                        -- Используется только для
                                        -- data-req-mode-time-period.
        data-req-person-id      INT-U16,        -- 0xFFFF доступно для всех лиц
        data-req-class          OID-Type,       -- Из раздела nom-part-obj
                                        -- подраздел MOC/BASE (MDC_MOC_VMD_*)
        data-req-obj-handle-list HANDLEList
}
-- Не назначенные значения «DataReqMode» сохраняются для
-- дальнейшего расширения и должны равняться нулю.
DataReqMode ::= BITS-16 {
        data-req-start-stop(0),         -- начать запрос данных: 1 | остановить
                                        --запрос данных: 0
        data-req-continuation(1),       -- продолжение запроса расчитанных с
                                        -- данных. Установить значение
                                        -- 1 для увеличения времени, выделенного
                                        -- для передачи данных. Если значение равно
                                        -- 1, то все другие биты игнорируются и
                                        -- следует использовать настройки команды
                                        -- первоначального запуска.
        -- следует установить один из следующих битов data-req-scope-*
        data-req-scope-all(4),
        data-req-scope-class(5),
        data-req-scope-handle(6),
        -- следует установить один из следующих битов data-req-mode-*
        data-req-mode-single-rsp(8),    -- ответ напрямую вкладывается в
                                        -- DataResponse
        data-req-mode-time-period(9),   -- запрос данных, ограниченный по
                                        -- времени,с ответами в виде уведомлений.
                                        -- Время указано в data-req-time в
                                        -- DataRequest.
        data-req-mode-time-no-limit(10), -- запрос данных,
```

```
                                    -- неограниченный по времени, с ответами
                                    -- в виде уведомлений.
        data-req-person-id(12)
}
DataReqModeCapab ::= SEQUENCE {
        data-req-mode-flags                 DataReqModeFlags,
        data-req-init-agent-count INT-U8,  -- максимальное количество
                                -- параллельных запросов/потоков данных от
                                -- агента.Должно иметь значение только от
                                -- 0 до 1.
        data-req-init-manager-count INT-U8 -- максимальное количество
                                -- параллельных запросов данных от менеджера
}
-- Не назначенные значения «DataReqModeFlags» сохраняются для
-- дальнейшего расширения и должны равняться нулю.
DataReqModeFlags ::= BITS-16 {    -- это поле используется в ходе
                                    -- ассоциации для обозначения
                                    -- параметров запросов данных
        data-req-supp-stop(0),      -- поддерживает прекращение текущего
                                    -- запроса данных
        data-req-supp-scope-all(4), -- поддерживает запрос всех
                                    -- объектов
        data-req-supp-scope-class(5), -- поддерживает запрос объектов
                                    -- на основе класса объекта
        data-req-supp-scope-handle(6), -- поддерживает запрос объектов
                                    -- по дескриптору объекта
        data-req-supp-mode-single-rsp(8), -- поддерживает одиночный
                                    -- запрос
        data-req-supp-mode-time-period(9), -- поддерживает запрос,
                                    -- ограниченный по времени
        data-req-supp-mode-time-no-limit(10), -- поддерживает запрос,
                                    -- неограниченный по времени
        data-req-supp-person-id(11),
        data-req-supp-init-agent(15)  -- агент использует
                                    -- запросы/потоки, задействованные
                                    -- агентом
}
-- Значение DataResponse возвращается в результате запроса MDS-Data-
-- Request (см. таблица 3). Однако поля event-type и event-info
-- заполняются с помощью параметров, указанных в события объекта
-- системы медицинского прибора. Значения допустимых типов событий
-- event-type, а также соответствующая структура event-info указаны
-- в таблице 4; в то же время для таких целей недопустимо
-- использование ConfigReport. Таким образом, значение event-info
-- должно быть одним из ScanReportInfoFixed, ScanReportInfoVar,
--ScanReportInfoMPFixed, или ScanReportInfoMPVar.
DataResponse ::= SEQUENCE {
        rel-time-stamp          RelativeTime,  -- установить 0xFFFFFFFF, если
                                    --не поддерживается RelativeTime
        data-req-result         DataReqResult,
        event-type              OID-Type,      -- event-type и event-info
                                    --только при условии
                                    -- data-req-mode-single-rsp, в ином
                                    -- случае event-type должен равняться 0
                                    -- и длина event-info = 0.
                                    -- Из раздела the nom-part-obj.
                                    -- Подраздел NOTI (MDC_NOTI_*)
        event-info              ANY DEFINED BY event-type
}
-- Значения в DataReqResult используются в поле DataResponse data-
```

```
-- req-result. Значение возвращается в ответ на DataRequest. Если
-- запрос был успешным, агент возвращает значение data-req-result-
-- no-error. В противном случае, возвращается одна из определенных
-- ошибок. Не назначенные значения «DataReqResult» сохраняются для
-- дальнейшего расширения и не используются.
DataReqResult ::= INT-U16 {
        data-req-result-no-error(0),
        data-req-result-unspecific-error(1),
        -- Следующие коды ошибок возвращаются, если запрос менеджера
        -- содержит DataReqMode, который не поддерживается агентом.
        data-req-result-no-stop-support(2),
        data-req-result-no-scope-all-support(3),
        data-req-result-no-scope-class-support(4),
        data-req-result-no-scope-handle-support(5),
        data-req-result-no-mode-single-rsp-support(6),
        data-req-result-no-mode-time-period-support(7),
        data-req-result-no-mode-time-no-limit-support(8),
        data-req-result-no-person-id-support(9),
        -- Следующие коды ошибок возвращаются, если запрос менеджера
        -- содержит неизвестные значения в дополнительных полях
        -- (например, data-req-person-id).
        data-req-result-unknown-person-id(11),
        data-req-result-unknown-class(12),
        data-req-result-unknown-handle(13),
        -- Следующие элементы указывают на случай, когда менеджер
        -- устанавливает более одного бита ограничения или режима.
        data-req-result-unsupp-scope(14),       -- установлены неподходящие
                                                -- биты ограничения
        data-req-result-unsupp-mode(15),        -- установлен неподходящий
                                                -- бит режима.
        data-req-result-init-manager-overflow(16),      -- менеджер
                                                -- попытался установить значение
                                                -- превышающее поток data-req-
                                                -- init-manager-count
        data-req-result-continuation-not-supported(17), -- менеджер
                                                -- попытался продолжить передачу
                                                -- данных, проводящуюся не в режиме
                                                -- ограниченного времени
        data-req-result-invalid-req-id(18)      -- менеджер попытался
                                                -- продолжить передачу данных на
                                                -- несуществующий data-req-id.
}
```

A.11.6 Сервисы сканера

Применительно к сервисам сканера используются определения сервисов системы медицинского прибора, перечисленные в разделе A.11.5, а именно

```
ScanReportInfoVar
ScanReportInfoFixed
ScanReportInfoGrouped
ScanReportInfoMPVar
ScanReportInfoMPFixed
ScanReportInfoMPGrouped
```

A.11.7 Типы данных, связанные с нумерацией

```
-- Простое наблюдаемое числовое значение представляет собой значение
-- с плавающей запятой.
--
SimpleNuObsValue ::= FLOAT-Type

-- Список типов SimpleNuObsValue
--
SimpleNuObsValueCmp ::= SEQUENCE OF SimpleNuObsValue
-- Во многих случаях, базовое наблюдаемое числовое значение
-- выражается меньшим значением с плавающей запятой.
```

```
--
BasicNuObsValue ::= SFLOAT-Type
-- Списковый тип данных BasicNuObsValue
--
BasicNuObsValueCmp ::= SEQUENCE OF BasicNuObsValue
```

А.11.8 Типы данных, связанные с PM-блоком и PM-сегментом

```
--
-- Атрибут PM-Store-Capab определяет специфические параметры и
-- свойства рабочей копии объекта PM-блока. По умолчанию значение
-- этого атрибута равно 0 (биты не заданы). Не назначенные значения
-- «PmStoreCapab» сохраняются для дальнейшего расширения и должны
-- равняться нулю.
--
PmStoreCapab ::=BITS-16 {
        pmsc-var-no-of-segm(0),         -- указывает, что число PM-сегментов
                                        -- в PM-блоке динамично и может меняться
        pmsc-epi-seg-entries(4), -- Некоторые/все PM-сегменты содержат
                                        -- эпизодические/апериодические записи и
                                        -- поэтому имеют подробную информацию о
                                        -- метке времени
        pmsc-peri-seg-entries(5), -- Некоторые/все PM-сегменты содержат
                                        -- периодически отбираемые записи и поэтому
                                        -- PM-сегмент или PM-блок должны
                                        -- поддерживать атрибут Sample-Period
        pmsc-abs-time-select(6), -- PM-сегменты в типе данных
                                        -- SegmSelection можно выбрать,
                                        -- назначив abs-time-range
        pmsc-clear-segm-by-list-sup(7), -- возможна очистка списка
                                        -- сегментов
        pmsc-clear-segm-by-time-sup(8), -- возможна очистка сегментов с
                                        -- сортировкой по времени
        pmsc-clear-segm-remove(9), -- если установлен этот бит, агент
                                        -- полностью удаляет указанную копию PM-
                                        -- сегмента в ходе применения метода Clear-
                                        -- Segment. Если этот бит не установлен, он
                                        -- лишь удалит все записи из выбранного
                                        -- PM-сегмента.
        pmsc-multi-person(12)   -- PM-блок позволяет использовать PM-
                                        -- сегмент нескольким лицам.
}

--
-- Все записи в сегменте должны соответствовать формату,
-- установленному данным атрибутом. Во-первых, необязательный
-- заголовок должен соответствовать описанию в segm-entry-header.
-- Таким образом, перед каждой записью будет находиться заголовок
-- (например, с указанием метки времени), применимый ко всем
-- элементам записи. Во-вторых, элементы должны соответствовать
-- формату и находиться в порядке, указанном в segm-entry-elem-list.
-- Обычно элемент содержит информацию об измерении. Хранимая
-- информация каждого элемента представлена в виде карты значения
-- атрибута (attribute value map), также как и метрических объектов.
--
PmSegmentEntryMap ::= SEQUENCE {
        segm-entry-header SegmEntryHeader,              -- определяет опциональные
                                        -- элементы перед каждой записью.
        segm-entry-elem-list SegmEntryElemList
}

--
-- Следующая строка битов определяет опциональные элементы данных
-- перед каждой записью сегмента. Многокомпонентные элементы данных
-- также могут быть определены. В этом случае, элемент данных с
```

```
-- меньшим номером бита располагаются перед элементами с большим
-- номером. Заголовок позволяет определить элементы данных
-- относящиеся ко всем элементам в записи. Если все биты имеют
-- нулевое значение, уведомление о записи сегмента должно начинаться
-- с данных из первого элемента. Не назначенные значения
-- «SegmEntryHeader» сохраняются для дальнейшего расширения и должны
-- равняться нулю. Если какой-либо из битов установлен помимо
-- запланированных битов (например, в новой версии был добавлен
-- новый бит), не следует производить восстановление данных,
-- поскольку невозможно вычислить отклонение от первого элемента
-- данных.
--
SegmEntryHeader ::= BITS-16 {
        seg-elem-hdr-absolute-time(0), --перед записью указывается
                                -- абсолютное время (тип данных
                                -- AbsoluteTime)
        seg-elem-hdr-relative-time(1), -- перед записью указывается
                                -- относительное время(тип данных
                                -- RelativeTime)
        seg-elem-hdr-hires-relative-time(2)-- перед записью указывается
                                -- высокоточное относительное время (тип
                                -- данных HighResRelativeTime)
}
SegmEntryElemList ::= SEQUENCE OF SegmEntryElem

--
-- Значение SegmEntryElem должно ссылаться на копию метрического
-- объекта в конфигурации агента с помощью значения его
-- идентификатора. Ссылаемый объект должен находиться в конфигурации
-- агента, а значения metric-type и class-id должны равняться
-- соответствующим атрибутам в указанном метрическом объекте.
--
SegmEntryElem ::= SEQUENCE {
        class-id        OID-Type, -- содержит номенклатурный код из
                                -- раздела OO nom-part-obj, определяющего
                                -- класс объекта (например, числовой)
        metric-type     TYPE, -- специфический статичный ТИП хранимого
                                -- элемента
        handle          HANDLE, -- идентификатор упоминаемого объекта
        attr-val-map    AttrValMap -- схема значений атрибута с описанием
                                -- хранимых данных
}
--
-- Для начала передачи указанного сегмента необходим запрос
--
TrigSegmDataXferReq ::= SEQUENCE {
        seg-inst-no             InstNumber
}
TrigSegmDataXferRsp ::= SEQUENCE {
        seg-inst-no             InstNumber,
        trig-segm-xfer-rsp      TrigSegmXferRsp
}
-- Не назначенные значения «TrigSegmXferRsp» сохраняются для
-- дальнейшего расширения и не используются.
TrigSegmXferRsp ::= INT-U16 {
        tsxr-successful(0),             -- Агент начнет передачу сегмента
        tsxr-fail-no-such-segment(1),   -- идентификатор сегмента не
                                -- обнаружен
        tsxr-fail-clear-in-process(2), -- идет очистка среды хранения
                                -- данных. Доступ запрещен
```

```
        tsxr-fail-segm-empty(3),        -- запрошенный сегмент пуст
        tsxr-fail-not-otherwise-specified(512)
}
--
-- SegmentDataEvent
--
-- П р и м е ч а н и е — Агент передает все записи сегментов в определенном
-- порядке, первая запись следует первой (порядок живой очереди).
--
SegmentDataEvent ::= SEQUENCE {
        segm-data-event-descr       SegmDataEventDescr,
        segm-data-event-entries     OCTET STRING  -- содержит указанные
                                    -- записи сегмента в непрозрачной структуре
                                    -- данных. Данное поле может содержать
                                    -- только полные записи.
}

SegmentDataResult ::= SEQUENCE {
        segm-data-event-descr SegmDataEventDescr
}

--
-- Segment Data Event Descriptor определяет, какая запись в данных
-- сегмента (Segment Data) связана с уведомлением.
--
SegmDataEventDescr ::= SEQUENCE {
        segm-instance       InstNumber, -- номер передаваемого
                            -- экземпляра сегмента
        segm-evt-entry-index  INT-U32, -- индекс массива первой
                            -- записи события
        segm-evt-entry-count  INT-U32, -- подсчет записей события
        segm-evt-status       SegmEvtStatus
}

-- Не назначенные значения «SegmEvtStatus» сохраняются для
-- дальнейшего расширения и должны равняться нулю.
SegmEvtStatus ::= BITS-16 {
        sevtsta-first-entry(0),         -- данное событие содержит первую
                                        -- запись сегмента
        sevtsta-last-entry(1),          -- данное событие содержит последнюю
                                        -- запись сегмента (может быть выбран
                                        -- как последний, так и первый бит,
                                        -- если все записи подходят к одному
                                        -- событию)
        sevtsta-agent-abort(4),         -- передача прервана агентом (менеджер
                                        -- в ответ должен отобразить
                                        -- аналогичный статус)
        sevtsta-manager-confirm(8),-- указывается в ответе, если
                                        -- сегмент успешно передан (если
                                        -- неуказан, агент повторяет последнее
                                        -- событие)
        sevtsta-manager-abort(12)  -- указывается в ответе менеджера
                                        -- (агент прекращает отправку
                                        -- сообщений)
}

SegmentStatistics ::= SEQUENCE OF SegmentStatisticEntry

SegmentStatisticEntry :: = SEQUENCE {
        segm-stat-type    SegmStatType,
        segm-stat-entry   OCTET STRING -- данный атрибут содержит одну
                                        -- запись сегментав формате,
                                        -- установленном PmSegmentEntryMap
}
```

```
-- Не назначенные значения «SegmStatType» сохраняются для
-- дальнейшего расширения и не используются. Значения от 0xF000 до
-- 0xFFFF сохраняются для расширений, ориентированных на
-- производителя.
        SegmStatType ::= INT-U16 {
        segm-stat-type-undefined (0),
        segm-stat-type-minimum(1),
        segm-stat-type-maximum(2),
        segm-stat-type-average(3)
```

Приложение В
(справочное)

Пример спецификации шкалы и диапазона

В.1 Общие положения

Алгоритм расчета шкалы и диапазона RT-SA описан в 6.3.5.3, а также приводится и здесь в качестве справочной информации:

$$Y = M \times X + B,$$

где Y — преобразованное абсолютное значение;

M = (верхнее-абсолютное-значение — нижнее-абсолютное-значение) / (верхнее-масштабируемое-значение — нижнее-масштабируемое-значение);

B = верхнее-абсолютное-значение — (M × верхнее-масштабируемое-значение);

X — масштабируемая величина.

Следует учитывать, что термин *абсолютное значение* не обозначает математическое абсолютное значение, в котором все значения положительны, а относится к текущему измеренному значению.

Формула позволяет заменить измеренное значение (с учетом диапазона смещения и ограниченного разрешения) на целочисленную скалярную величину, которая сможет уменьшить количество информации, передаваемой между агентом и менеджером. Структуры ScaleRangeSpec8, 16 и 32, описанные в А.3.4, передают оба верхнее и нижнее абсолютные значения, а также верхнее и нижнее масштабируемые значения, позволяя менеджеру определить параметры формулы для конвертирования масштабируемых значений в соответствующие абсолютные значения и подтвердить, что полученные значения в ожидаемом диапазоне.

В рамках агента масштабируемое значение, получаемое из реального измеренного значения, рассчитывается следующим образом:

$$X = (R — B) / M,$$

где R — реальное измеренное значение.

Походящее значение M позволяет обеспечить масштабируемые значения, чтобы передать подходящее разрешение для абсолютных измеренных значений. Практически, параметры M и B могут быть заданы разрешением АЦП и прочими техническими средствами.

В.2 Пример термометра

Ниже приводится пример алгоритма. Показания термометра, способного определять температуру по шкале Цельсия от –45 °С до 50 °С с разрешением 0.5 °С передаются в виде беззнаковых значений с помощью ScaleRangeSpec8.

Следующие значения применимы для структуры ScaleRangeSpec8:

Нижнее-абсолютное-значение = –45.0

Верхнее-абсолютное-значение = 50.0

Нижнее-масштабируемое-значение = 0

Верхнее-масштабируемое-значение = 190

Следовательно

$M = (50.0 – (–45.0))/(190 – 0) = 0.5$

$B = 50.0 – (0.5 \times 190) = –45.0$

Некоторые характерные значения приведены в таблице В.1, а на рисунке В.1 отображена диаграмма масштабированных и конвертированных значений.

Таблица В.1— Схема преобразования

Масштабированное (x)	Конвертированное (y)
0	–45.0
50	–20.0
100	5.0
150	30.0
190	50.0

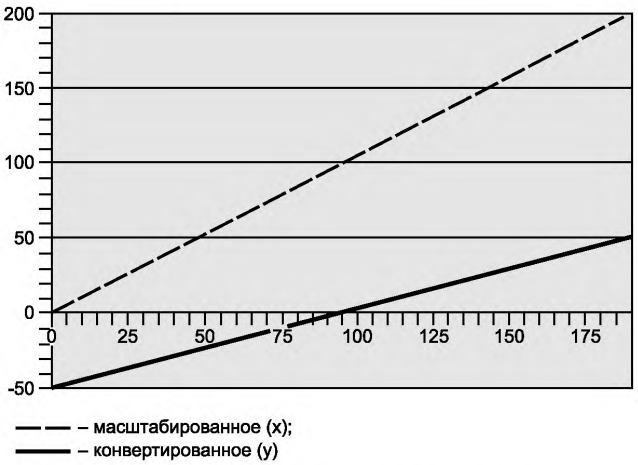


Рисунок В.1 — Графический вид преобразования

Приложение С
(справочное)

Концепция РМ-блока

С.1 Общие положения

Концепция РМ-блока предлагает метод отображения, доступа и передачи большого количества метрических данных, хранящегося в агенте. Информация хранится в виде многоуровневой модели объекта с возможностью структурирования данных в соответствии с их характером.

На высшем уровне объекты РМ-блока являются точкой первичного доступа ко всей информации о хранящихся метрических данных. Агент, поддерживающий постоянно хранимые метрические данные, может создавать экземпляры одного или нескольких объектов РМ-блока. Объект РМ-блока является частью конфигурации устройства. Прямой доступ к нему можно получить с помощью сервисов доступа к объекту, описанных в данном стандарте.

Каждый РМ-блок может хранить 0, 1 или несколько РМ-сегментов, которые являются контейнерными объектами фактических данных. В ходе работы агента количество РМ-сегментов может меняться. Другими словами, агент может создавать новые РМ-сегменты с учетом временных интервалов, размера хранимых данных или ручного управления пользователем.

Концепция РМ-блока предлагает информационную модель с двухуровневой иерархией с несколькими объектами РМ-сегментов внутри нескольких объектов РМ-блока.

К типичным вариантам использования нескольких РМ-блоков относятся следующие случаи. Если агент хранит данные с различными параметрами (например, апериодические измерения в сравнении с периодическими измерениями), то отдельные объекты РМ-блока используются для назначения оптимизированных типов данных для хранимой информации и, таким образом, экономит память для хранимых данных.

К типичным вариантам использования нескольких РМ-сегментов относятся следующие случаи. Если агенту необходимо структурировать хранимую информацию в более иерархичной форме, он может использовать несколько экземпляров объектов РМ-блока с экземплярами объектов РМ-сегментов для построения такой иерархии (например, можно задействовать РМ-блок для отображения сеанса обучения, а затем использовать РМ-сегмент для моделирования индивидуальных упражнений в рамках сеанса обучения).

Для хранения фактических данных используется концепция схемы значений атрибута для метрических атрибутов. Особый отображающий атрибут позволяет определить структуру двоичных хранимых данных, не допуская излишних затрат ресурсов на идентификацию, размер полей и т.д. в фактических хранимых и передаваемых двоичных данных. Предполагается, что хранимая информация представляет собой большой массив одинаково форматированных данных.

После проверки информации в объектах РМ-блока, менеджер начинает передачу хранимых данных. Менеджер может выбрать фрагменты данных для передачи. Фактическая передача считается завершенной после получения подтверждающего уведомления со стороны агента. Агент должен максимально полно заполнить структуру данных SegmentDataEvent.

С.2 Постоянная иерархия объектов хранения метрических данных

С.2.1 Общая информация

Постоянное метрическое хранилище состоит из следующих четырех ключевых частей:

- РМ-блок. Данный объект относится к высшему уровню и содержит атрибуты об объектах хранения, а также ноль или несколько РМ-сегментов;
- РМ-сегмент.Данный объект содержит атрибуты, описывающие сегмент, а также ноль или несколько записей;
- Запись.Каждая запись содержит опциональный заголовок и один или несколько элементов;
- Элемент. Каждый элемент содержит данные одного или нескольких измерений.

На рисунке С.1 приведено примерное устройство этих четырех компонентов. Более подробную информацию см. в конце данного приложения.

С.2.2 Объект РМ-блока

Поддержка объекта РМ-блока является опциональной. Поддержка объектов, атрибутов, методов и соответствующих событий РМ-блока необходима лишь тем агентам, которые хотят хранить постоянные метрические данные. Менеджеру следует сообщить информацию обо всех поддерживаемых объектах РМ-блока, входящих в конфигурацию агента. Атрибуты РМ-блока описывают общие характеристики хранимых данных (например, является ли хранение значений периодическим или эпизодическим).

Агент может позволять использовать несколько РМ-блоков. Множество блоков используется для отображения данных в разных форматах или с различными характеристиками, а также для группировки данных по различным логическим группам.Объект РМ-блока также доступен для всех методов, связанных с хранимыми метрическими данными (в частности, с восстановлением данных менеджером).

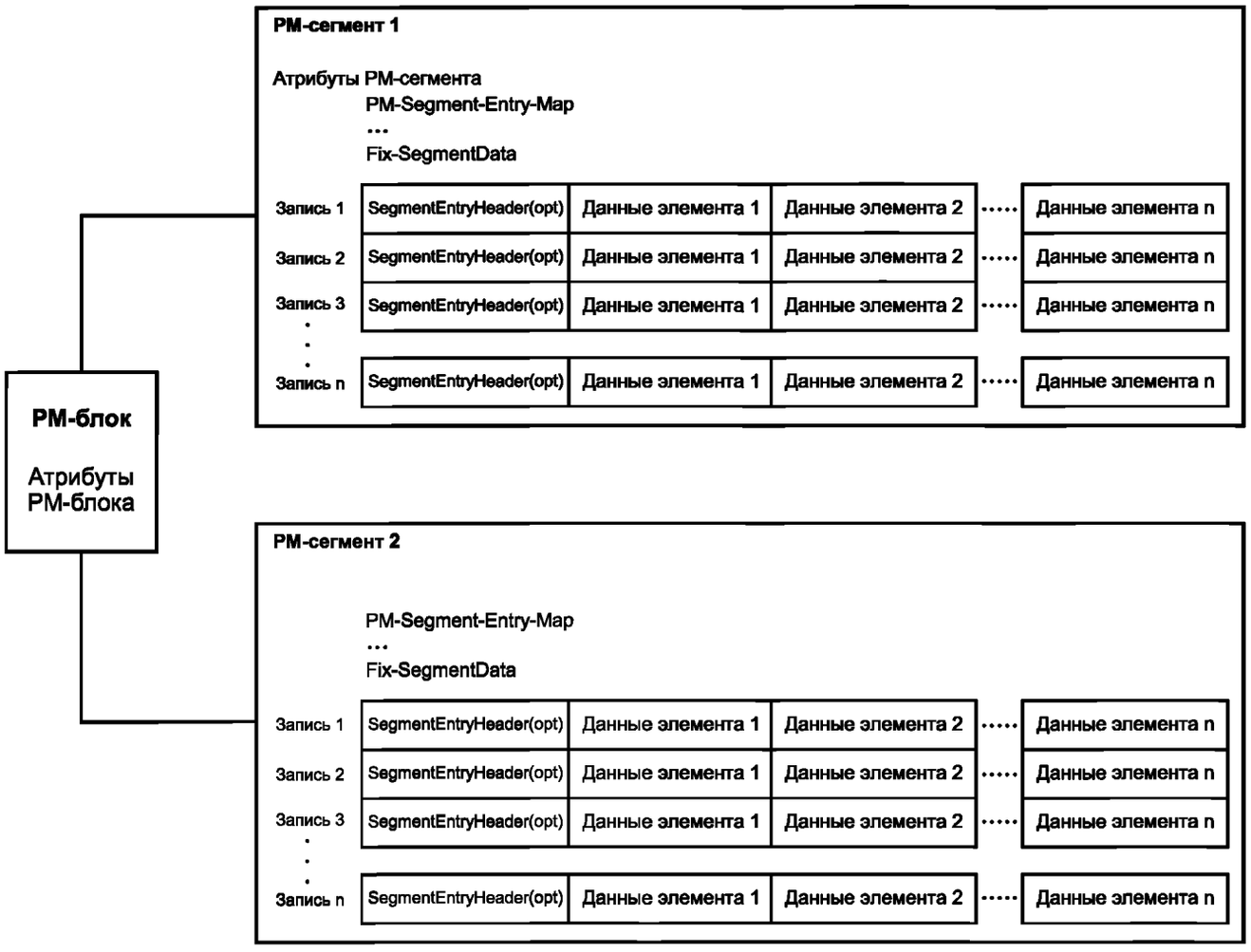


Рисунок С.1 — PM-блок с 2 PM-сегментами и fixed-segment-data в сегментах

Агент может управлять количеством PM-сегментов, существующих в PM-блоке. При отсутствии данных, считается, что PM-блок не содержит элементов. Если же есть какие-либо данные, PM-блок содержит один или несколько объектов PM-сегмента. Поскольку количество PM-сегментов является динамическим, объекты PM-сегмента не входят в конфигурацию агента. Вместо этого, объект PM-блока содержит информацию о доступных PM-сегментах в форме атрибутов PM-блока, запрошенных через сервис GET.

С.2.3 Объект PM-сегмента

Базовый формат данных сегмента PM-сегмента показан на рисунке С.2.

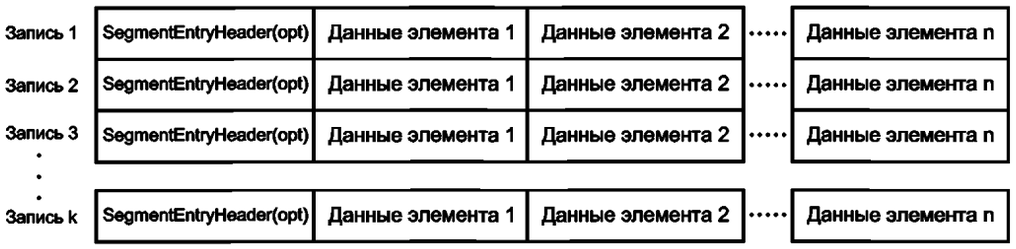


Рисунок С.2 — Формат данных PM-сегмента

Сегмент содержит *k* записей. Формат записи определяется атрибутом PM-Segment-Entry-Map PM-сегмента.

В записи отображаются данные, хранимые в определенный момент времени. Перед каждой записью стоит опциональный заголовок (например, с отметкой времени) общий для всех элементов записи. Далее в записи содержится n элементов; формат каждого элемента определяется схемой значений атрибутов. Обычно запись содержит результат измерения (например, численное измерение и нумерацию). Итоговая структура данных не содержит каких-либо идентификаторов атрибута или длину поля и потому является очень компактной.

PM-сегмент обычно представляет собой эпизод для хранения. Данный эпизод имеет временной контекст (то есть, к примеру, данные, хранящиеся в этом эпизоде, принадлежат к отрезку времени 12:00—15:00), некоторые соответствующие атрибуты и массив памяти, содержащий фактические (измеренные) данные для этого эпизода, содержащиеся в атрибуте Fixed-Segment-Data (см. рисунок С.3).

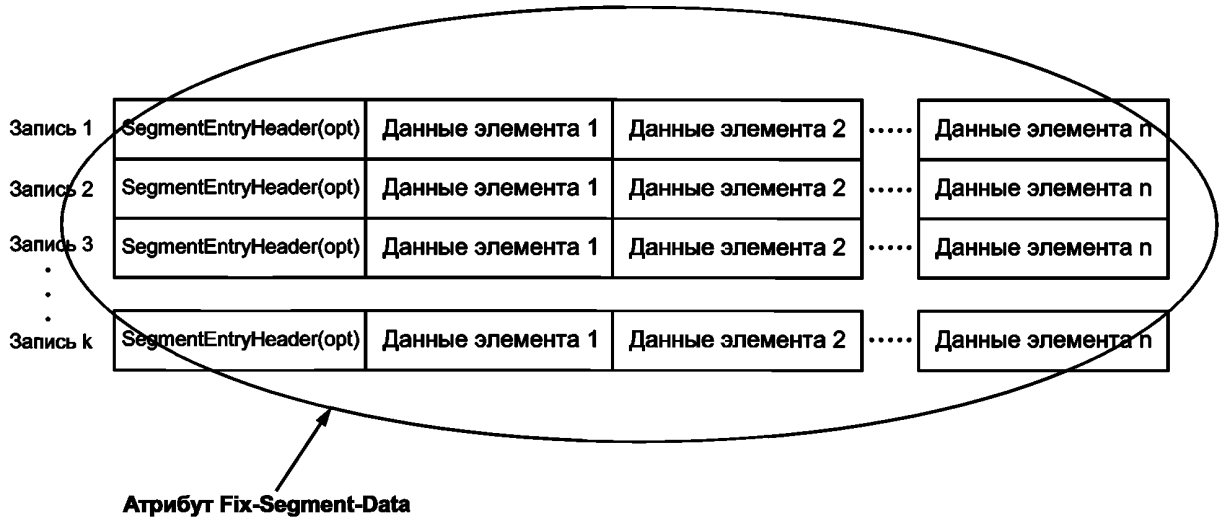


Рисунок С.3 — Атрибут Fixed-segment-data, содержащий фактические хранимые данные

PM-блок может содержать несколько PM-сегментов или ни одного (т. е. ни одного, если данные еще не сохранены; один или несколько в зависимости от хранимых эпизодов и возможностей агента).

К примеру, PM-сегмент действующих часов может содержать сохраненные данные об одном тренировочном цикле (т. е. 5-минутный отрезок времени с 12:00). Устройство также может хранить несколько сегментов (т. е. несколько таких тренировочных циклов).

С.2.4 Запись PM-сегмента (в рамках fixed-segment-data)

Атрибут Fixed-Segment-Data включает в себя как записи, так и элементы. На рисунке С.4. группы записи отображены в виде строк. Все записи в сегменте обладают структурой, определяемой PM-Segment-Entry-Map. Это, в свою очередь, сопоставимо с Attribute-Value-Map, определенным для метрических объектов. Однако такая структура позволяет группировать несколько измерений в подгруппу записи.



Рисунок С.4 — Запись (элемент массива в fixed-segment-data)

Атрибут PM-Segment-Entry-Map определяет список измерений, хранящихся в одной записи. Для каждого измерения также назначается список хранящихся атрибутов. Кроме того, при желании можно назначить общий заголовок (т.е. с включением общей отметки времени) для всей записи.

Используя приведенный выше пример с работающими часами, предположим, что агент хранит данные о частоте сердечных сокращений, текущей скорости бега и значении насыщенности периферийным кислородом в секунду. Единственным атрибутом, сохраненным из этих измерений, будет числовым значением (определяемым PM-Segment-Entry-Map). В таком случае, заголовок записи не требуется, поскольку измерения периодичны и не требуют наличия отметки времени. В частности, в случаях периодических измерений, время конкретного сохранен-

ного измерения рассчитывается из времени начала и окончания измерения, пробного периода и индекса записи. Вследствие этого, наличие отдельной отметки времени для каждого измерения не требуется, равно как и заголовок с информацией об отметке времени.

Таким образом, каждая строка записи включает в себя следующие три компонента:

ЧСС	Скорость	Насыщенность периферийным кислородом
120	10	98

С.2.5 Входные группы записи PM-сегмента

Группа содержит двоичное представление определенных атрибутов одного метрического объекта (см. рисунок С.5). SegmEntryElem (смотрите пункт А.11.8) в рамках PM-Segment-Entry-Map определяет входную группу каждой записи.



Рисунок С.5 — Элемент. Набор атрибутов одного измерения

В примере с работающими часами в записи смоделировано три метрических значения. Для каждого метрического значения определяется один атрибут, числовое наблюдаемое значение. Таким образом, значения частоты сердечных сокращений, скорости и насыщенности периферийным кислородом являются отдельными группами внутри записи.

Однако PM-Segment-Entry-Map может содержать атрибуты вне наблюдаемого значения. К примеру, можно включить такие атрибуты как достоверность, отметки времени, коды блока и так далее.

Приложение D
(справочное)

Типы транспортного профиля

D.1 Общие положения

Настоящий стандарт применяет концепцию «типа» для группирования и определения сервисов, предлагаемых различными технологиями передачи данных и предназначенные для семейства стандартов ИСО/ИИЭР 11073. В частности, семейство стандартов ИСО/ИИЭР 11073 различает следующие типы профилей передачи:

Тип 1. Транспортные профили, содержащие **одновременно** надежные и наилучшие транспортные сервисы. Используется один или несколько виртуальных каналов для надежных средств передачи, а также один или несколько виртуальных наилучших транспортных сервисов.

Тип 2. Транспортные профили, содержащие **только** однонаправленный транспортный сервис.

Тип 3. Транспортные профили, содержащие только максимально доступный транспортный сервис. Используется один или несколько виртуальных каналов наилучших транспортных сервисов.

Типы транспортных профилей имеют большое значение, поскольку различные транспортные сервисы влияют на функциональность верхних уровней системы. В частности, они влияют на работу механизма подтверждающих сервисов данного стандарта.

В термине «механизм подтверждающего сервиса» слово *подтверждающий* имеет следующее значение:

- для данных (сервис уведомлений EVENT REPORT), информирующий агента о том, когда менеджер «принял ответственность» за часть данных и агент может удалить эту величину;
- для управления (сервисы ACTION, GET и SET), информирующий менеджера о том, когда агент «завершил» запрошенную передачу.

D.2 Тип 1

Транспортный профиль Типа 1 предоставляет как надежные, так и наилучшие транспортные профили. Учитывая определение и цели механизма подтверждающего сервиса, потеря пакетов данных сильно влияет на подтвержденные сообщения. Таким образом, самым подходящим транспортным сервисом для подтвержденных сообщений будет надёжный транспортный сервис.

Кроме того, согласно данному стандарту (см. пункт 8.4) машины состояний агента и менеджера синхронизированы. Наличие синхронизированных машин состояний косвенно подразумевает использование надежного транспортного сервиса между двумя машинами состояний, обеспечивающего доставку сообщения и уведомляющего о сбое в ходе его транспортировки.

Таким образом, все сообщения, связанные с процессом ассоциации, передаются с помощью надежного транспортного сервиса. (Для простоты поиска машины состояний агента и менеджера, указанные в пункте 8.4, будут называться машинами состояний, перекликаясь с машинами состояний транспортного профиля Типа 1). Для неподтвержденных сообщений прикладное программное обеспечение может, по своему усмотрению, использовать либо надежный, либо наилучший транспортный сервис (см. таблицу D.1).

Таблица D.1 — Использование транспортного профиля Типа 1

Транспортный сервис	Сообщения ИИЭР 11073-20601	
	Процедура ассоциации & Подтверждено	Не подтверждено
Наилучший	Не поддерживается	Поддерживается
Надежный	Поддерживается	Поддерживается

Транспортный профиль, относящийся к профилю Типа 1, должен поддерживать один или несколько надежных виртуальных каналов, либо ни одного или несколько наилучших виртуальных каналов.

D.3 Тип 2

Транспортный профиль Типа 2 предлагает только однонаправленный транспортный сервис. Учитывая определение и цели механизма подтверждающего сервиса, однонаправленный транспортный сервис непригоден для подтвержденных сообщений (Менеджер не может отправлять подтвержденные сообщения обратно агенту).

Однонаправленный сервис, по своей сути, является наилучшим сервисом. Транспортный уровень менеджера не может сделать запрос на повторную передачу транспортного уровня, если потерян протокольный блок

данных (ПБД). Таким образом, надежный транспортный сервис несовместим с однонаправленным транспортным сервисом.

В связи с отсутствием надежного транспортного сервиса конечный автомат Типа 1 не сможет корректно взаимодействовать с транспортным профилем Типа 2. Таким образом, для среды Типа 2 необходим конечный автомат Типа 2 (см. таблицу D.2).

Таблица D.2 — Использования транспортного профиля Типа 2

Транспортный сервис	Сообщения ИИЭР 11073-20601	
	Процедура ассоциации и Подтверждено	Не подтверждено
Наилучший	Не поддерживается	Поддерживается
Надежный	Не поддерживается	Не поддерживается

D.4 Тип 3

D.4.1 Общие положения

Транспортный профиль Типа 3 предлагает только наилучший транспортный сервис. Отсутствие надежного транспортного сервиса вызывает определенные трудности с использованием конечного автомата Типа 1 и механизма подтверждаемого сервиса. Существует несколько разных, не взаимоисключающих методов решения таких проблем.

D.4.2 Тип 3a

Одним из методов разрешения такой ситуации является добавление дополнительной функции надежного транспортного сервиса к наилучшему сервису. После применения такого метода, транспортный профиль Типа 2 (только наилучший (best-effort-only), по своей сути, станет транспортным профилем Типа 1 (надежный и наилучший). В таком случае возможно использование конечного автомата Типа 1 и механизма подтверждаемого сервиса.

Таким образом, транспортный профиль Типа 3a считается особым случаем транспортного профиля Типа 1.

D.4.3 Тип 3b

Еще одним способом использования транспортного сервиса best-effort-only является перемещение функциональности надежного транспортного сервиса в протокол медицинского прибора. Результатом данного метода является конечный автомат Типа 3 и механизм подтверждающего сервиса Типа 3 (см. таблица D.3).

Таблица D.3 — Использование транспортного профиля Типа 3b

Транспортный сервис	Сообщения ИИЭР 11073-20601	
	Процедура ассоциации и Подтверждено	Не подтверждено
Наилучший	Поддерживается	Поддерживается
Надежный	Не поддерживается	Не поддерживается

D.4.4 Тип 3c

Третьим методом использования транспортного сервиса best-effort-only является добавление дополнительной функции облегченной надежности (reliability-lite) в транспортный сервис best-effort. Данный метод похож на Типа 3a. Однако в транспортном сервисе облегченной надежности Типа 3c некоторые характеристики надежного транспортного сервиса несколько ослаблены. Планируется, что такое ослабление будет способствовать более компактной и простой реализации с облегченной надежностью по сравнению с полностью надежным транспортным сервисом.

Ослабленные фактические характеристики надежного транспортного сервиса позволят определить корректность работы конечного автомата Типа 1 и механизма подтверждения сервиса в среде транспортного профиля Типа 3c

D.5 Вывод

Все типы транспортных профилей отображены в таблице D.4.

Таблица D.4 — Типы транспортных профилей

Транспортный профиль	Описание	Вид «2 x 2»			Конечный автомат ассоциированный и подтвержденный	Режимы передачи данных
Тип 1/3a	Надежный и наилучший		Подт.	Неподт.	Тип 1	3
		Наилучший	НЕТ	ok		
		Надежный	ok	ok		
Тип 2	Только однонаправленный		~~Подт.~~	Неподт.	Новый Тип 2	1
		Наилучший	НЕТ	ok		
		Надежный	НЕТ	НЕТ		
Тип 3b	Только наилучший		Подт.	Неподт.	Новый Тип 3	2
		Наилучший	ok	ok		
		Надежный	НЕТ	НЕТ		
Тип 3c	Облегченной надежности и наилучший		Подт.	Неподт.	Не определен (возможно Тип 1)	3
		Наилучший	НЕТ	ok		
		Облегченной надежности	ok	ok		

Приложение Е
(обязательное)

Таблицы состояний

Е.1 Общие положения

Все состояния, отображаемые в таблице состояний агента и менеджера, приведены в таблице Е.1.

Таблица Е.1 — Состояния

Номер состояния	Состояние	Используется агентом	Используется менеджером
1	Отключено	ДА	ДА
2	Подключено Не ассоциировано	ДА	ДА
3	Подключено Ассоциируется	ДА	
4	Подключено Ассоциировано Настраивается Отправка настроек	ДА	
5	Подключено Ассоциировано Настраивается Ожидание Подтверждения	ДА	
6	Подключено Ассоциировано Настраивается Ожидание		ДА
7	Подключено Ассоциировано Настраивается Проверка настроек		ДА
8	Подключено Ассоциировано Выполнение	ДА	ДА
9	Подключено Разъединение	ДА	ДА

Е.2 Таблица состояний агента

Конечный автомат агента используется в соответствии с таблицей Е.2, в которой используются следующие обозначения:

REQ — запрос от прикладного программного обеспечения (ПО), взаимодействующего через интерфейс с конечным автоматом;

IND — состояние, утвержденное нижним уровнем ПО через конкретный программный интерфейс;

Rx — пакет данных протокола прикладного уровня, полученный через поток входных данных;

Tx — пакет данных протокола прикладного уровня, отправленный через поток выходных данных.

Таблица Е.2 — Таблица состояний агента

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
1.1	Отключено	IND транспортного соединения	Подключено Не ассоциировано	«Shall» указывает на уровень применения	Отсутствует
2.2	Подключено Не ассоциировано	IND транспортного разрыва	Отключено	«Should» указывает на уровень применения	Отсутствует
2.5	Подключено Не ассоциировано	REQ Assoc	Подключено Ассоциируется	Timeout=reset, retry=reset.	Tx aarq

Продолжение таблицы Е.2

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
2.6	Подключено Неассоциировано	REQ AssocRel	Подключено Не ассоциировано <нет смены состояний>	Отсутствует	Отсутствует
2.7	Подключено Не ассоциировано	REQ AssocAbort	Подключено Не ассоциировано <нет смены состояний>	Недопустимо	Tx abrt
2.8	Подключено Не ассоциировано	Rx aarq(*)	Подключено Не ассоциировано	Ассоциация агент-агент	Tx aare (постоянно отклоненный)
2.12	Подключено Не ассоциировано	Rx aare(*)	Подключено Не ассоциировано <нет смены состояний>	Недопустимо	Tx abrt
2.16	Подключено Не ассоциировано	Rx rlrq	Подключено Не ассоциировано <no state transition>	Недопустимо	Tx abrt
2.17	Подключено Не ассоциировано	Rx rlre	Подключено Не ассоциировано <нет смены состояний>	Недопустимо Игнорировать	Отсутствует
2.18	Подключено Не ассоциировано	Rx abrt	Подключено Не ассоциировано <нет смены состояний>	Отсутствует	Отсутствует
2.19	Подключено Не ассоциировано	Rx prst(*)	Подключено Не ассоциировано <нет смены состояний>	Недопустимо	Tx abrt
3.2	Подключено Ассоциируется	IND транспортного соединения	Отключено	Отсутствует	Отсутствует
3.3	Подключено Ассоциируется	IND тайм-аута, не достигнуто максимальное количество повторных попыток	Подключено Ассоциируется <нет смены состояний>	Таймер=сброс, повтор++	Tx aarq
3.4	Подключено Ассоциируется	IND тайм-аута, достигнуто максимальное количество повторных попыток	Подключено Не ассоциировано	Отсутствует	Tx abrt
3.6	Подключено Ассоциируется	REQ AssocRel	Подключено Не ассоциировано	Отсутствует	Tx abrt
3.7	Подключено Ассоциируется	REQ AssocAbort	Подключено Не ассоциировано	Отсутствует	Tx abrt
3.8	Подключено Ассоциируется	Rx aarq(*)	Подключено Не ассоциировано	Ассоциация агент-агент	Tx aare (rejectedperm anent)

Продолжение таблицы Е.2

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
3.13	Подключено Ассоциируется	Rx Aare (accepted)	Подключено Ассоциировано Работает	Указывает на случаи прямого перехода в рабочее состояние	Отсутствует
3.14	Подключено Ассоциируется	Rx aare (accepted -unknown-config)	Подключено Ассоциировано Настраивается Отправка настроек	Менеджер принял ассоциацию, но не имеет настроек	Отсутствует
3.15	Подключено Ассоциируется	Rx aare(rejected-*)	Подключено Не ассоциировано	Отсутствуют дальнейшие попытки подключения	Отсутствует
3.16	Подключено Ассоциируется	Rx rlrq	Подключено Не ассоциировано	Недопустимо. Агент получил запрос на ассоциацию, но еще не установил ее	Tx abrt
3.17	Подключено Ассоциируется	Rx rlre	Подключено Не ассоциировано	Недопустимо	Tx abrt
3.18	Подключено Ассоциируется	Rx abrt	Подключено Не ассоциировано	Отсутствует	Отсутствует
3.19	Подключено Ассоциируется	Rx prst(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
4.2	Подключено Ассоциировано Настраивается Отправка настроек	IND транспортного разрыва	Отключено	Отсутствует	Отсутствует
4.4	Подключено Ассоциировано Настраивается Отправка настроек	IND Timeout	Подключено Не ассоциировано	Нет ответа	Tx abrt
4.6	Подключено Ассоциировано Настраивается Отправка настроек	REQ AssocRel (*)	Подключено Разъединяется	ПО запрашивает завершение ассоциации. Тайм-аут=сброс (Timeout=reset)	Tx rlrq(*)
4.7	Подключено Ассоциировано Настраивается Отправка настроек	REQ AssocAbort	Подключено Не ассоциировано	Преждевременное завершение программного обеспечения	Tx abrt
4.8	Подключено Ассоциировано Настраивается Отправка настроек	Rx aarq(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
4.12	Подключено Ассоциировано Настраивается Отправка настроек	Rx aare	Подключено Не ассоциировано	Недопустимо	Tx abrt

Продолжение таблицы Е.2

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
4.16	Подключено Ассоциировано Настраивается Отправка настроек	Rx rlrq	Подключено Не ассоциировано	Отсутствует	Tx rlre(normal)
4.17	Подключено Ассоциировано Настраивается Отправка настроек	Rx rlre	Подключено Не ассоциировано	Недопустимо	Tx abrt
4.18	Подключено Ассоциировано Настраивается Отправка настроек	Rx abrt	Подключено Не ассоциировано	Отсутствует	Отсутствует
4.22	Подключено Ассоциировано Настраивается Отправка настроек	Rx roiv-cmip-get, handle=0	Подключено Ассоциировано Настраивается Отправка настроек <нет смены состояний>	Менеджеру разрешено изучение системы медицинского прибора. См. 6.3.2.6.1	rors-cmip-get. (атрибуты системы медицинского прибора)
4.23	Подключено Ассоциировано Настраивается Отправка настроек	Rx roiv-* but not (roivcmip-get, handle=0)	Подключено Ассоциировано Настраивается Отправка настроек <нет смены состояний>	Запрещено, пока не достигнуто рабочее состояние	Tx roer (nosuchobject-instance)
4.26	Подключено Ассоциировано Настраивается Отправка настроек	Rx (rors-*, roer, or rorj)	Подключено Не ассоциировано	Недопустимо	Tx abrt
4.32	Подключено Ассоциировано Настраивается Отправка настроек	REQ Send (ConfigReport)	Подключено Ассоциировано Настраивается Ожидание подтверждения	Конфигурация агента еще не опробована менеджером	Tx EventReport (Config Report)
5.2	Подключено Ассоциировано Настраивается Ожидание подтверждения	IND сбоя передачи	Отключено	Отсутствует	Отсутствует
5.4	Подключено Ассоциировано Настраивается Ожидание подтверждения	IND тайм-аута	Подключено Не ассоциировано	Нет ответа	Tx abrt
5.6	Подключено Ассоциировано Настраивается Ожидание подтверждения	REQ AssocRel(*)	Подключено Разъединяется	Подключено Разъединение	Tx rlrq(*)

Продолжение таблицы Е.2

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
5.7	Подключено Ассоциировано Настраивается Ожидание подтверждения	REQ AssocAbort	Подключено Не ассоциировано	Сбой программного обеспечения	Tx abrt
5.8	Подключено Ассоциировано Настраивается Ожидание подтверждения	Rx aarq(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
5.12	Подключено Ассоциировано Настраивается Ожидание подтверждения	Rx aare(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
5.16	Подключено Ассоциировано Настраивается Ожидание подтверждения	Rx rlrq	Подключено Не ассоциировано	Отсутствует	Tx rlre(normal)
5.17	Подключено Ассоциировано Настраивается Ожидание подтверждения	Rx rlre	Подключено Не ассоциировано	Недопустимо	Tx abrt
5.18	Подключено Ассоциировано Настраивается Ожидание подтверждения	Rx abrt	Подключено Не ассоциировано	Отсутствует	Отсутствует
5.22	Подключено Ассоциировано Настраивается Ожидание подтверждения	Rx roiv-cmipget, handle=0	Подключено Ассоциировано Настраивается Отправка настроек	Менеджеру разрешено изучение системы мед. прибора. См. раздел 6.3.2.6.1	rors-cmip-get. (атрибуты системы медицинского прибора)
5.23	Подключено Ассоциировано Настраивается Ожидание подтверждения	Rx roiv-* but not (roivcmip-get, handle=0)	Подключено Ассоциировано Настраивается Отправка настроек	Запрещено, пока не достигнуто рабочее состояние	Tx roer (nosuchobject-instance)
5.27	Подключено Ассоциировано Настраивается Ожидание подтверждения	Rx rors-cmipconfirmedevent-report (unsupportedconfig)	Подключено Ассоциировано Настраивается Отправка настроек	Менеджер отклонил конфигурацию	Отсутствует
5.29	Подключено Ассоциировано Настраивается Отправка настроек	Rx rors-cmip-confirmed-event-report (accepted-config)	Подключено Ассоциировано Выполнение	Менеджер принял конфигурацию	Отсутствует

Продолжение таблицы Е.2

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
5.30	Подключено Ассоциировано Настраивается Ожидание подтверждения	Rx (rors-*, roer, or rorj), but not Rx: rors-cmip-confirmed-event-report	Подключено Не ассоциировано	Недопустимо	Tx abrt
8.2	Подключено Ассоциировано Выполнение	IND сбоя передачи	Отключено	Отсутствует	Отсутствует.
8.4	Подключено Ассоциировано Выполнение	IND тайм-аута	Подключено Не ассоциировано	Нет ответа	Tx abrt
8.6	Подключено Ассоциировано Выполнение	REQ AssocRel	Подключено Разъединение.	Отсутствует Timeout=reset	Tx rlrq(normal)[5]
8.7	Подключено Ассоциировано Выполнение	REQ AssocAbort	Подключено Не ассоциировано	Отсутствует	Tx abrt
8.8	Подключено Ассоциировано Выполнение	Rx aarq(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
8.12	Подключено Ассоциировано Выполнение	Rx aare(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
8.16	Подключено Ассоциировано Выполнение	Rx rlrq	Подключено Не ассоциировано	Если у агента есть особые идентификаторы вызова invoke-ids, предполагается, что он не получит ответ на свой запрос	Tx rlre(normal)
8.17	Подключено Ассоциировано Выполнение	Rx rlre	Подключено Не ассоциировано	Недопустимо	Tx abrt
8.18	Подключено Ассоциировано Выполнение	Rx abrt	Подключено Не ассоциировано	Отсутствует	Отсутствует
8.21	Подключено Ассоциировано Выполнение	Rx roiv-*	Подключено Ассоциировано Выполнение <нет смены состояний>	Нормальная передача сообщений. Это нормальный режим Выполнения	Tx (rors-*, or roer, or rorj)
8.26	Подключено Ассоциировано Выполнение	Rx (rors-*, roer, or rorj)	Подключено Ассоциировано Выполнение <нет смены состояний>	Нормальная передача сообщений. Это нормальное состояние выполнения	Отсутствует[6]

[5] AssocRel отправляется только после того, как особые идентификаторы вызова invoke-ids устареют.

[6] Если rors-* получены с неизвестным идентификатором вызова, прикладной уровень вызовет отправку менеджеру уведомление о прерывании, отправив значения "REQ abrt" конечному автомату.

Окончание таблицы Е.2

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
9.2	Подключено Разъединение	IND сбоя передачи	Отключено	Отсутствует	Отсутствует
9.4	Подключено Разъединение	IND тайм-аута	Подключено Не ассоциировано	Нет ответа	Tx abrt
9.6	Подключено Разъединение	REQ AssocRel	Подключено Разъединение	Уже разъединяется Игнорировать	Отсутствует
9.7	Подключено Разъединение	REQ AssocAbort	Подключено Не ассоциировано	Отмена постепенного процесса разъединения	Tx abrt
9.8	Подключено Разъединение	Rx aarq	Подключено Не ассоциировано	Недопустимо	Tx abrt
9.12	Подключено Разъединение	Rx aare(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
9.16	Подключено Ассоциировано Работает	Rx rlrq	Подключено Разъединение <нет смены состояний>	Обе стороны выполняют рассоединение. Ответ и ожидание собственного rlre	Tx rlre(normal)
9.17	Подключено Разъединение	Rx rlre	Подключено Не ассоциировано	Процесс рассоединения завершен, переход к неассоциированному состоянию	Отсутствует
9.18	Подключено Разъединение	Rx abrt	Подключено Не ассоциировано	Отсутствует	Отсутствует
9.21	Подключено Разъединение	Rx roiv-*	Подключено Разъединение <нет смены состояний>	Менеджер отправил сообщения вызова, когда агент отправил rlrq. Агент вышел из состояния Выполнение и, следовательно, не даст ответа	Отсутствует
9.26	Подключено Разъединение	Rx (rors-*, roer, or rorj)	Подключено Не ассоциировано	***Примеры 1 Прикладной уровень имеет не подтвержденные идентификаторы вызова (invoke-ids), но в любом случае уже сформировал ReleaseRequest. 2 Непредусмотренный rors-****	Tx abrt

Е.3 Таблица состояний менеджера

Конечный автомат менеджера используется в соответствии с таблицей Е.3, в которой присутствуют следующие обозначения:

REQ — запрос от прикладного ПО, взаимодействующего через интерфейс с машиной состояний;

IND — состояние, утвержденное нижним уровнем ПО через конкретный программный интерфейс;
Rx — пакет данных протокола прикладного уровня, полученный через поток входных данных;
Tx — пакет данных протокола прикладного уровня, отправленный через поток выходных данных.

Таблица Е.3 — Таблица состояний менеджера

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
1.1	Отключено	IND Транспортное соединение	Подключено Не ассоциировано	«Shall» указывает на уровень приложения	Отсутствует
2.2	Подключено Не ассоциировано	IND Разрыв передачи	Отключено	“Should” указывает на уровень приложения	Отсутствует
2.6	Подключено Не ассоциировано	REQ AssocRel	Подключено Не ассоциировано <нет смены состояний>	Недопустимо Игнорировать	Отсутствует
2.7	Подключено Не ассоциировано	REQ AssocAbort	Подключено Не ассоциировано <нет смены состояний>	Недопустимо Игнорировать	Tx abrt
2.9	Подключено Не ассоциировано	Rx aarq (допустимо и известна конфигурация)	Подключено Ассоциировано Работает	Устройство и конфигурация известны менеджеру	Tx aare (accepted)
2.10	Подключено Не ассоциировано	Rx aarq (допустимо с неизвестной конфигурацией)	Подключено Ассоциировано Настраивается Ожидание	Менеджер определяет, что соединение приемлемо, но не обладает действительной информацией о конфигурации для агента	Tx aare (accepted-unknown-config)
2.11	Подключено Не ассоциировано	Rx aarq (недопустимая конфигурация)	Подключено Не ассоциировано	Менеджер определяет, что соединение недопустимо	Tx aare (reject-*)
2.12	Подключено Не ассоциировано	Rx aare(*)	Подключено Не ассоциировано <нет смены состояний>	Недопустимо	Tx abrt
2.16	Подключено Не ассоциировано	Rx rlrq	Подключено Не ассоциировано <нет смены состояний>	Недопустимо	Tx abrt
2.17	Подключено Не ассоциировано	Rx rlre	Подключено Не ассоциировано <нет смены состояний>	Недопустимо Игнорировать	Отсутствует
2.18	Подключено Ассоциируется	Rx abrt	Подключено Не ассоциировано <нет смены состояний>	Недопустимо Игнорировать	Отсутствует
2.19	Подключено Не ассоциировано	Rx prst(*)	Подключено Не ассоциировано <нет смены состояний>	Недопустимо	Tx abrt

Продолжение таблицы Е.3

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
6.2	Подключено Ассоциировано Настраивается Ожидание	IND Разрыв передачи	Отключено	Отсутствует	Отсутствует
6.4	Подключено Ассоциировано Настраивается Ожидание	IND Timeout	Подключено Не ассоциировано	Нет ответа	Tx abrt
6.6	Подключено Ассоциировано Настраивается Ожидание	REQ AssocRel	Подключено Разъединение	Отсутствует Timeout=reset	Tx rlrq (normal)
6.7	Подключено Ассоциировано Настраивается Ожидание	REQ AssocAbort	Подключено Не ассоциировано	Отсутствует	Tx abrt
6.8	Подключено Ассоциировано Настраивается Ожидание	Rx aarq(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
6.12	Подключено Ассоциировано Настраивается Ожидание	Rx aare(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
6.16	Подключено Ассоциировано Настраивается Ожидание	Rx rlrq	Подключено Не ассоциировано	Менеджер получил запрос на сброс ассоциации	Tx rlre (normal)
6.17	Подключено Ассоциировано Настраивается Ожидание	Rx rlre	Подключено Не ассоциировано	Недопустимо	Tx abrt
6.18	Подключено Ассоциировано Настраивается Ожидание	Rx abrt	Подключено Не ассоциировано	Отсутствует	Отсутствует
6.24	Подключено Ассоциировано Настраивается Ожидание	Rx roivconfirmede-ventreport	Подключено Ассоциировано Настраивается Проверка настроек	Уведомление содержит конфигурацию, полученную от агента	Отсутствует
6.26	Подключено Ассоциировано Настраивается Ожидание	Rx (rors-*, roer, или rorj)	Подключено Ассоциировано Настраивается Ожидание <нет смены состояний>	Менеджер может отправить roiv-cmip-get (handle=0). См.6.3.2.6.1	Отсутствует[7)]

[7)] Если rors-* получены с неизвестным идентификатором вызова, уровень применения отправит агенту уведомление о прерывании с помощью отправки значения “REQ abrt” машине состояний.

Продолжение таблицы Е.3

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
7.2	Подключено Ассоциировано Настраивается Проверка настроек	IND Разрыв передачи	Отключено	Отсутствует	Отсутствует
7.4	Подключено Ассоциировано Настраивается Проверка настроек	IND тайм-аут	Подключено Не ассоциировано	Нет ответа	Tx abrt
7.6	Подключено Ассоциировано Настраивается Проверка настроек	REQ AssocRel	Подключено Разъединение	Отсутствует	Tx rlrq (normal)
7.7	Подключено Ассоциировано Настраивается Проверка настроек	REQ AssocAbort	Подключено Не ассоциировано	Отсутствует	Tx abrt
7.8	Подключено Ассоциировано Настраивается Проверка настроек	Rx aarq(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
7.12	Подключено Ассоциировано Настраивается Проверка настроек	Rx aare(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
7.16	Подключено Ассоциировано Настраивается Проверка настроек	Rx rlrq	Подключено Не ассоциировано	Менеджер получил запрос на сброс ассоциации	Tx rlre (normal)
7.17	Подключено Ассоциировано Настраивается Проверка настроек	Rx rlre	Подключено Не ассоциировано	Недопустимо	Tx abrt[8)]
7.18	Подключено Ассоциировано Настраивается Проверка настроек	Rx abrt	Подключено Не ассоциировано	Отсутствует	Отсутствует
7.24	Подключено Ассоциировано Настраивается Проверка настроек	Rx roiv-confirmed-event-report	Подключено Ассоциировано Настраивается Проверка настроек <нет смены состояний>	Агент отправляет измерения, прежде чем согласовать конфигурацию	Если нет отчета о конфигурации, то Tx roer (no-such-object-instance). Если есть отчет о конфигурации, то Tx abrt

[8)] Если rors-* получены с неизвестным идентификатором вызова, уровень применения отправит агенту уведомление о прерывании с помощью отправки значения“REQ abrt” машине состояний.

Продолжение таблицы Е.3

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
7.25	Подключено Ассоциировано Настраивается Проверка настроек	Rx roiv-* но не (roiv-confirmed-event-report)	Подключено Не ассоциировано	Агент отправляет только сообщения отчетов о событии. Этого не должно никогда происходить	Tx roer (no-such-action)
7.26	Подключено Ассоциировано Настраивается Проверка настроек	Rx (rors-*, roer, или rorj)	Подключено Ассоциировано Настраивается Проверка настроек <нет смены состояний>	Менеджер мог отправить roiv-cmip-get (handle=0). См. 6.3.2.6.1	Отсутствует
7.31	Подключено Ассоциировано Настраивается Проверка настроек	REQ агент предоставил неподдерживаемую конфигурацию	Подключено Ассоциировано Настраивается Ожидание	Отсутствует	Tx rors-cmip-configuration-event (unsupported-config)
7.32	Подключено Ассоциировано Настраивается Проверка настроек	REQ агент предоставил неподдерживаемую конфигурацию	Подключено Ассоциировано Работает	Отсутствует	Tx rors-cmip-configuration-event (supported-config
8.2	Подключено Ассоциировано Выполнение	IND Разрыв передачи	Отключено	Отсутствует	Отсутствует
8.4	Подключено Ассоциировано Выполнение	IND тайм-аут	Подключено Не ассоциировано	Нет ответа	Tx abrt
8.6	Подключено Ассоциировано Выполнение	REQ AssocRel	Подключено Разъединение	Отсутствует	Tx rlrq (normal) или Tx rlrq (configuration-changed)[9)]
8.7	Подключено Ассоциировано Выполнение	REQ AssocAbort	Подключено Не ассоциировано	Отсутствует	Tx abrt
8.8	Подключено Ассоциировано Выполнение	Rx aarq(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
8.12	Подключено Ассоциировано Выполнение	Rx aare(*)	Подключен Не ассоциировано	Недопустимо	Tx abrt
8.16	Подключено Ассоциировано Выполнение	Rx rlrq	Подключено Не ассоциировано	Отсутствует	Tx rlre(normal)
8.17	Подключено Ассоциировано Выполнение	Rx rlre	Подключено Не ассоциировано	Недопустимо	Tx abrt

[9)] AssocRel отправляется только после того, как особые идентификаторы вызова invoke-ids устареют.

Продолжение таблицы Е.3

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
8.18	Подключено Ассоциировано Выполнение	Rx abrt	Подключено Не ассоциировано	Отсутствует	Отсутствует
8.21	Подключено Ассоциировано Выполнение	Rx roiv-*	Подключено Ассоциировано Выполнение <нет смены состояний>	Нормальная передача сообщений. Это нормальный режим выполнения	Tx (rors-*, or roer, или rorj)
8.26	Подключено Ассоциировано Выполнение	Rx (rors-*, roer, или rorj)	Подключено Ассоциировано Выполнение <нет смены состояний>	Нормальная передача сообщений. Это нормальный режим выполнения	Отсутствует[10)]
9.2	Подключено Разъединение	IND Разрыв передачи	Отключено	Отсутствует	Отсутствует
9.4	Подключено Разъединение	IND тайм-аут	Подключено Не ассоциировано	Нет ответа	Tx abrt
9.6	Подключено Разъединение	REQ AssocRel	Подключено Разъединение <нет смены состояний>	Уже разъединяется. Игнорировать	Отсутствует
9.7	Подключено Разъединение	REQ AssocAbort	Подключено Не ассоциировано	Отмена постепенного процесса разъединения	Tx abrt
9.8	Подключено Разъединение	Rx aarq	Подключено Не ассоциировано	Недопустимо	Tx abrt
9.12	Подключено Разъединение	Rx aare(*)	Подключено Не ассоциировано	Недопустимо	Tx abrt
9.16	Подключено Разъединение	Rx rlrq	Подключено Разъединение <нет смены состояний>	Обе стороны выполняют рассоединение. Ожидание собственного rlre	Tx rlre (normal)
9.17	Подключено Разъединение	Rx rlre	Подключено Не ассоциировано	Процесс рассоединения завершен, переход к неассоциированному состоянию	Отсутствует
9.18	Подключено Разъединение	Rx abrt	Подключено Не ассоциировано	Отсутствует	Отсутствует
9.21	Подключено Разъединение	Rx roiv-*	Подключено Разъединение <нет смены состояний>	Агент отправил сообщения вызова, когда менеджер отправил rlrq. Менеджер вышел из состояния Выполнение и, следовательно, не даст ответа	Отсутствует

[10)] Если rors-* получены с неизвестным идентификатором вызова, уровень применения отправит агенту уведомление о прерывании с помощью отправки значения “REQ abrt” машине состояний.

Окончание таблицы Е.3

ID сигнала	Исходное состояние	Событие/ входной поток	Следующее состояние	Семантическое поведение/ примечание	Tx поток (выходной)/ сгенерирован событием
9.26	Подключено Разъединение	Rx (rors-*, roer, or rorj)	Подключено Не ассоциировано	***Примеры*** ***1 Прикладной уровень имеет не подтвержденные идентификаторы вызова (invoke-ids), но в любом случае уже сформировал ReleaseRequest.*** ***2 Непредусмотренный rors-****	Tx abrt[11)]

[11)] Если rors-* получены с неизвестным идентификатором вызова, уровень применения отправит агенту уведомление о прерывании с помощью отправки значения “REQ abrt” машине состояний.

Приложение F
(обязательное)

Правила кодирования медицинских приборов (MDER)

F.1 Общие положения

Настоящее приложение заимствовано из ИСО/ИИЭР 11073-20101:2004 [14], А.1—А.4. Они представлены для удобства реализации.

Настоящее приложение определяет специализированные правила MDER, связанные с представлением последовательных двоичных строк, таким образом, каким они должны быть представлены в сети в сравнении с соответствующей организацией в памяти компьютера, с представлением в абстрактном синтаксисе, то есть в языке программирования или диаграмм, которые используются в спецификациях. Предполагается, что данная спецификация должна быть согласована с любой и каждой из альтернатив нижнего уровня ИСО/ИИЭР 11073. Таким образом, реализация на верхних уровнях может обеспечивать прозрачность на основе конкретного профиля нижнего уровня.

Основные цели MDER включают в себя возможность оптимизировать выполнение форматирования и синтаксического анализа, а также снижают нагрузку на пропускную способность сети. Оптимизация форматирования основана на возможности процессора передачи данных определять так называемые сообщения с фиксированным форматом (canned), в которые только динамически изменяемые данные должны быть включены для относительно высокочастотных сообщений, в особенности это касается данных осцилограмм.

F.2 Поддерживаемый синтаксис ASN.1

ASN.1 является стандартным обозначением, которое используется для определения типов данных, значений и ограничений значений. Данное обозначение широко используется в стандартах ВОС и широко используется в серии стандартов ИСО/ИИЭР 11073 (например, в ИСО/ИИЭР 11073-10201, где все определения данных формируются с помощью ASN.1).

В целях выполнения требований эффективности кодирования и декодирования и поддержки сообщений с фиксированным форматом, MDER определяет методы для преобразования синтаксиса ASN.1 в поток байтов, пригодный для передачи.

В отличие от других стандартов ИСО/ВОС для правил кодирования ASN.1 (например, основные правила кодирования или BER, правила компактного кодирования или PER) правила MDER оптимизированы только для подмножества ASN.1. Правила MDER не поддерживают полный набор типов данных ASN.1, а лишь определенный, ограниченный набор конструкций ASN.1.

Стандарты ИСО/ИИЭР 11073 используют этот ограниченный набор ASN.1 для определения типов данных, применяемых только в управляемых медицинских объектах, таким образом, правила MDER подходят и являются достаточными для кодирования структур данных в рамках этих стандартов.

Ограниченный набор ASN.1, используемый для компонентов PDU стандартов ИСО/ИИЭР 11073, является строгим подмножеством допустимых типов данных ASN.1, поэтому другие общие стандартные правила кодирования (например, BER, PER) можно так же использовать, как согласованные для конкретного профиля коммуникаций на более высоких уровнях.

Таблица F.1 определяет специализацию ASN.1, подходящую для кодирования MDER. Все компоненты ASN.1 PDU, предназначенные для кодирования MDER, являются предметами этой специализации.

Для каждого типа данных ASN.1 эта специализация сопровождает символом«I» для включенного с ограничением, «R» для ограничений по использованию и «E» для исключения.

Таблица F.1 — Типы данных, поддерживаемые ASN.1

Тип ASN.1	Статус	Комментарии
INTEGER	R	Целочисленный тип. Размерные ограничения должны быть использованы для всех типов данных INTEGER, для определения диапазона значений целого числа. Краткие имена для поддерживаемых типов ограничений определяются следующим образом: INT-U8 ::= INTEGER(0..255) INT-I8 ::= INTEGER (-127..128) INT-U16 ::= INTEGER (0..65535) INT-I16 ::= INTEGER (-32768..32767) INT-U32 ::=INTEGER (0..4294967295) INT-I32 ::= INTEGER (-2147483648..2147483647) Только сокращенные, ограниченные по размеру типы данных INTEGER следует использовать с определениями типов данных для кодирования в MDER

Продолжение таблицы F.1

Тип ASN.1	Статус	Комментарии
BIT STRING	R	Битовая строка. Размерные ограничения должны быть использованы для всех типов данных BIT STRING, для определения диапазона значений битовой строки. Краткие имена для поддерживаемых типов ограничений определяются следующим образом: BITS-8 :: = BIT STRING (SIZE(8)) BITS-16 :: = BIT STRING (SIZE(16)) BITS-32 :: = BIT STRING (SIZE(32)) Только сокращенные, ограниченные по размеру типы данных BIT STRING следует использовать с определениями типов данных для кодирования в MDER
OCTET STRING	I	—
SEQUENCE	R	Может не использовать следующие типы тегирования: OPTIONAL (опциональный), DEFAULT (по умолчанию), автоматический
SEQUENCE OF	I	—
CHOICE	R	Выбор. Может использоваться явная и неявная индексация
ANY DEFINED BY	I	ПРОИЗВОЛЬНЫЙ ТИП должен определять компонент в структуре данных (в основном в SEQUENCE), который определяет структуру этих данных для преобразователя кода/синтаксического анализатора (парсера)

F.3 Порядок передачи байтов

На рисунке F.1 показано, как различные двоичные строки сети отображаются в строки памяти. На диаграммах представлен порядок передачи байтов в сети (Network byte order, NBO). Следующие правила пронумерованы для удобства использования ссылок:

1) представление в диаграммах использует формат NBO, показанный на рисунке F.1;

2) в MDER не используется выравнивание. То есть в строки байтов дополнительные байты не добавляются, например, для получения длин, которые делятся на два или четыре. Тем не менее, переменная длина элементов данных, то есть строк, должна содержать четное число байтов из соображений эффективности. Например, поскольку большинство элементов данных 16-битные, они не будут неправильно выровненными, если строки имеют четную длину;

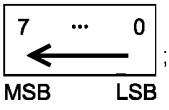
3) передачи данных в MDAP ограничены использованием соглашения NBO (обратный порядок передачи);

4) для обеспечения общей интероперабельности протокол ассоциации должен использовать ИСО BER при согласовании условных обозначений MDER. Все остальные блоки PDU, которыми обменивается в период своей работы хост-устройство, будут основаны на MDER, например PDU CMIP* и ROSE*. Суффикс звездочка (*) означает, что MDER используется для оптимизации протокола ИСО, который, как правило, базируется на BER.

Многобайтовые структуры отображаются между сетью и компьютерной памятью и упорядочиваются в памяти компьютера двумя основными способами, называемыми *big endian* (формат с порядком следования байтов, начиная со старшего), и *little endian* (формат с порядком следования байтов, начиная с младшего). Формат big endian согласуется с NBO, а little endian — не согласуется. Например, в последнем примере на рисунке F.1, структура ABCD была бы упорядочена как DCBA. В этом случае, если big endian является согласованным протоколом, то компьютер с little endian должен был бы переставлять компоненты этой структуры при получении их из памяти и передаче их в память, в случае необходимости. Макросы языка программирования и команды компьютера, выполняющие байтовый свопинг, которые, как правило, способствуют нормализации, являются проблемами реализации и могут быть упрощены ненормативными определениями, взятыми из настоящего стандарта или стандартов, связанных с ним.

NBO:

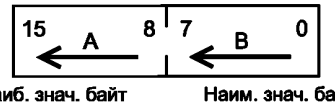
- Однобайтовая строка битов, т. е. октета:
  - Последовательность битов: в порядке от наименее значащего бита (LSB) к наиболее значащему биту (MSB), т. е. 0, ...,7 или 24, ..., 31; порядок следования битов представляется диаграммами посредством последующих обозначений ←, в которых конец стрелки обозначает последний переданный бит:



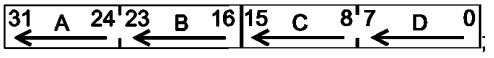
- Многобайтовая строка:
  - Неструктурированная: массив октет (т. е. строка октет):
  - Последовательность битов: для каждого типа, как определено для октеты;
    - Последовательность байтов: в основном нумеруется от [0] до [n-1], например A[0], до A[n-1], где <n>= длина в октетах



  - Структурированная: многобайтовая последовательность битов, в основном кратные двум байтам (например, короткое целое число — 16 битов, длинное целое число — 32 бита); числа с плавающей точкой в основном являются кратными 16 битам, хотя в настоящем стандарте определен только 32-х битный формат FLOAT. Приведены два общих примера (ABCD относится к порядку передачи байтов):
  - 16 битовая структура, например, короткое (целое число):
    - Последовательность битов: каждый байт пересылается согласно определению для октеты;
    - Последовательность байтов: пересылается от наиболее значащего байта к наименее значащему байту;
    - Для целых чисел со знаком обычно MSB наиболее значащего байта — знаковый бит



  - 32-битная структура, например, длинное (целое число)



  - Условно, составы мультиструктуры показаны в порядке появления в последовательно передаваемой строке

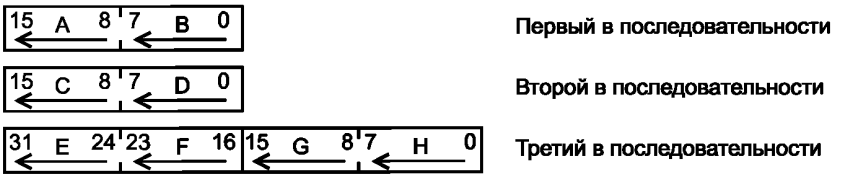


Рисунок F.1 — NBO — соглашения представления двоичной строки

F.4 Кодирование

F.4.1 Общие положения

В MDER отсутствует тегирование для простых типов. Теги используются только там, где дешифратору (декодеру) необходимо различать типы (например, для типа CHOICE). Поля длины используются только для элементов с переменной длиной и ограничены 64 КБайтами (16 битами), которых должно быть достаточно для передачи данных.

Простые типы определены из-за ограничений по размеру и имеют фиксированную длину.

Типы SEQUENCE имеют фиксированную длину, так как синтаксические компоненты типа OPTIONAL не используются.

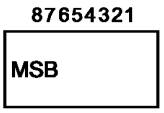
F.4.2 Тип INTEGER

Кодирование целочисленного значения является примитивным кодированием, а содержимое октет представляет значение в дополнительном двоичном коде.

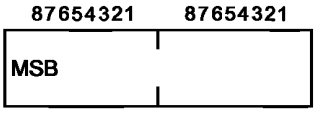
На рисунке F.2 представлено поддерживаемое в MDER кодирование октет[12)] для ограниченных по размеру целочисленных значений.

[12)] Для стандартизации языка программирования С для целочисленных типов данных, необходимо использовать определения ИСО/МЭК 9899.

– 8-битовые типы INT-U8, INT-I8



– 16-битовые типы INT-U16, INT-I16



– 32-битовые типы INT-U32, INT-I32

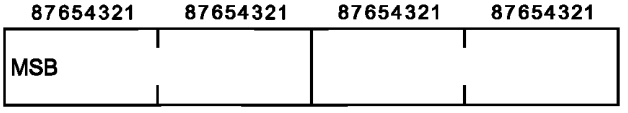


Рисунок F.2 — Кодирование целых чисел

Октеты содержат закодированные целочисленные значения в дополнительном двоичном коде.

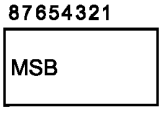
F.4.3 Тип BIT STRING

Кодирование значения битовой строки, относящейся к базовому типу, является простым. Содержимое октеты представляет множество битов в битовой строке. Битовая строка может содержать 8, 16 или 32 бита.

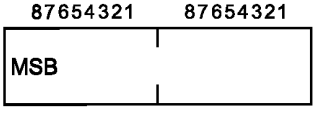
Бит 0 при кодировании является самым старшим битом (MSB), бит 1 представлен следующим битом в октете и т. д.

На рисунке F.3 представлено поддерживаемое в MDER кодирование октет для ограниченных по размеру битовых строк.

– 8-битовые типы BITS 8



– 16-битовые типы BITS16



– 32-битовые типы BITS 32

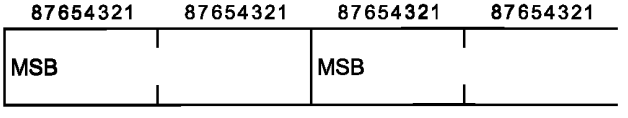


Рисунок F.3 — Кодирование битовой строки

Пример — Определение

state ::= BITS-16 {open(0), locked(1) }

может быть отображено на представление типа языка С следующим образом:

short unsigned int state;
#define locked 0x4000
#define open 0x8000

(подобно для именованных битов в в строках бит).

F.4.4 Тип OCTET STRING

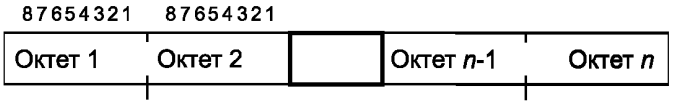
Кодирование значения OCTET STRING, относящегося к базовому типу, является простым. Содержимое октетов представляет собой строку элементов. Сами октеты используют кодирование, унаследованное от определения типа строки.

Октеты могут содержать печатаемые символы ASCII или могут содержать инкапсулированные двоичные данные. OCTET STRING, содержащие печатаемые символы ASCII, должны содержать четное число октет и ис-

пользовать символ NULL для дополнения. Необходимо отметить, что строки, которые содержат четное число октет, могут не завершаться символом NULL.

Как показано на рисунке F.4, MDER различают тип OCTET STRING с переменной длиной строки и тип OCTET STRING с фиксированной длиной строки.

- Фиксированный (ограниченный по размеру) тип: OCTET STRING ((SIZE(*n*))



- Типы OCTET STRING переменной длины

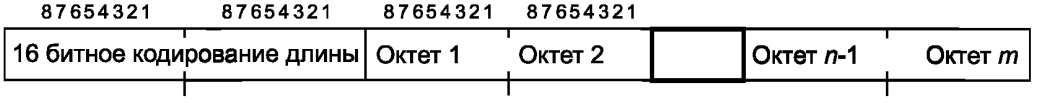


Рисунок F.4 — Кодирование типов OCTET STRING

Тип OCTET STRING с фиксированной (т. е. ограниченной по размеру) длиной строки кодируется только с соответствующим набором октет.

Типы OCTET STRING переменной длины кодируются полем с длиной 16 бит (целое число без знака в дополнительном двоичном коде), за которым следует определенное число октет с данными.

Пример — Следующие определения

```
fixed-sized-label        ::= OCTET STRING (SIZE(12))
    variable-label       ::= OCTET STRING
```

могут быть отображены в представлении типа языка С следующим образом:

```
typedef unsigned char fixed_size_label[12];
typedef struct {
    unsigned short length;
    unsigned char data[1];        /* Это место для заполнения подходящим по размеру*/
                                  /* массивом соответствующей длины */
} variable_label;
```

F.4.5 Тип SEQUENCE

Кодирование значения списка типа (SEQUENCE) формируется кодированием каждого элемента SEQUENCE в порядке их определения в типе ASN.1 SEQUENCE. Никакое выравнивание не выполняется.

Пример — Следующие определения

```
IdentType ::= SEQUENCE {
    id INT-U16,
    instanceINT-U16
    }
```

могут быть отображены на представление типа языка С следующим образом:

```
typedef struct {
   unsigned shortid,
   unsigned shortinstance
   }  IdentType;
```

и кодирование по MDER будет иметь вид, представленный на рисунке F.5.

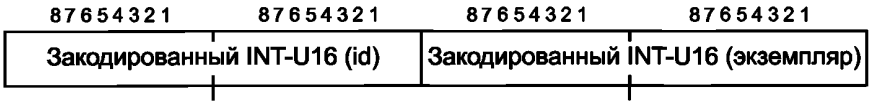


Рисунок F.5 — Образец кодирования типа SEQUENCE

F.4.6 Тип SEQUENCE OF

Кодирование значения SEQUENCE OF конструируется, а октеты содержания представляют закодированные значения элементов типа SEQUENCE OF, таким образом, чтобы ему предшествовало поле счетчика, указывающее на число элементов, и поле длины, указывающее полную длину структуры данных (в которой не учитываются сами счетчик и длина). За заголовком следуют упорядоченные закодированные элементы. См. рисунок F.6.

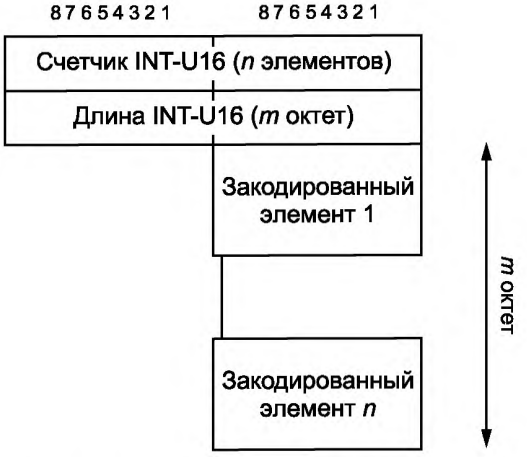


Рисунок F.6. — Кодирование типа SEQUENCE OF

Поле счетчика и поле длины с содержанием «0» указывает на пустой список структуры данных. Такая комбинация значений допускается.

Пример — Следующие описания:

```
Array1 ::= SEQUENCE OF Entry
```

могут быть отображены на представление типа языка С следующим образом:

```
typedef struct {
        unsigned short      count;
        unsigned short      length;
        Entry                       data[1]; /* placeholder for sufficient
number of entries */
} Array1;
```

F.4.7 Тип CHOICE

Кодирование значения выбора конструируется, а октеты содержания представляют закодированные значения выбранной альтернативы, таким образом, чтобы ему предшествовало поле тега, указывающее на выбранную альтернативу, и поле длины, указывающее длину кодирования выбранной альтернативы. См. рисунок F.7.

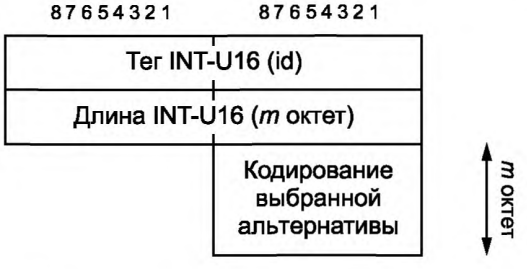


Рисунок F.7 — Кодирование типа CHOICE

Пример — Следующее описание

```
ChoiceType ::= CHOICE {
        one             OneType,
        two             TwoType
}
```

может быть отображено в представлении типа языка С следующим образом:

```
typedef struct {
        unsigned short      choice_id;
        unsigned short      length;
        union {
                            OneType         one;
                            TwoType         two;
        } data;
} ChoiceType;
#define one_type_chosen 1
#define two_type_chosen 2.
```

- Правила для значений тегов определяются следующим образом:
- теги могут быть заданы явно или неявно;
- абстрактный синтаксис для неявно заданных тегов не содержит явно заданного номера выбора и, следовательно, требуется правило для присвоения значений полю choice_id. Для неявно заданных тегов значения поля choice_id начинаются со значения 1 и последовательно размещаются в порядке абстрактных синтаксических выборов. В приведенном выше примере, значения поля choice_id для полей one_type_chosen и two_type_chosen будут равны 1 и 2 соответственно;
- абстрактный синтаксис для явно заданных тегов содержит явно заданный номер выбора, который отражается непосредственно в поле choice_id только что определенного правила кодирования. В этом случае процедуры выбора должны выполняться последовательно и в зависимости от применения могут быть несвязными, как показано в следующем примере:

```
choice-type ::= CHOICE {
        one [1] OneType,-- defines tag value 1 in MDER
        four [4] FourType -- defines tag value 4 in MDER
}
```

F.4.8 Тип ANY DEFINED BY и Instance-of

ANY DEFINED BY (ASN.1 1988/90) или тип instance-of (ASN.1 1994) кодируется заголовком поля длины, чтобы задать число октет в кодировании выбранного значения, как представлено ниже. См. рисунок F.8.

Данный тип ссылается на встроенные синтаксисы, которые задаются с помощью зарегистрированного идентификатора объекта (OID). См. приложение Н ИСО/ИИЭР 11073-20101 [14] для случаев совместимости.



Рисунок F.8 —Кодирование ANY DEFINED BY (instance-of)

Пример — Следующее описание

```
TestType ::= SEQUENCE {
        type-id                 OIDType,
        value                   ANY DEFINED BY type-id
}
```

может быть отображено в представление типа языка С следующим образом:

```
typedef struct {
        OIDType             type-id,
        unsigned short      any_length;
        char                any_data; /* поле для заполнения данными */
                                      /* закодированного типа*/
} TestType;
```

Данный пример показывает, кодирование байтов SEQUENCE, содержащее контекстно зависимый идентификатор объекта, и значение ANY DEFINED BY.

В предыдущем преобразовании, поле type-id является контекстно-свободным идентификатором объекта. Приложение должно использовать поле идентификатора, чтобы привести поле any_data к правильному типу дан-

ных. Символьный тип данных для поля any_data, по существу, не несет значения и предоставляет только адрес поля. Следует отметить, что длина может быть 0, что означает, что поле any_data не существует.

Тип Instance-of кодирует ASN.1 конструкцию TYPE-IDENTIFIER и идентичен ANY DEFINED BY, кодирующим с целью обратной совместимости.

F.5 Числа с плавающей точкой

Ограниченное подмножество ASN.1, которое может быть отображено с MDER не содержит тип данных FLOAT. Вместо этого, в настоящем стандарте определены собственные типы данных с плавающей точкой FLOAT-Type и SFLOAT-Type.

F.6 Структура данных с плавающей точкой FLOAT-Type

Тип FLOAT-Type отображается как 32-битная структура, форматированная в соответствии с цифровым форматом медицинских приборов (MDNF).

MDNF это 32-битное слово, содержащее 8-битную целочисленную экспоненту со знаком, за которой следует 24 битная целочисленная мантисса со знаком. См. рисунок F.9.



Рисунок F.9 — Кодирование MDNF

Представленное число будет иметь вид: (мантисса)×$10^{\text{экспонента}}$. И экспонента и мантисса записываются в двоичном дополнительном коде. Для обозначения точности мантисса и экспонента выравниваются, как описано в F.8.

Существует специальные значения, которые представлены в таблице F.2.

Таблица F.2 — Специальные значения MDNF

Специальное значение	Мантисса	Битовое значение
NaN (не число)	+ (2^{23} – 1)	0x007FFFFF
NRes (не при таком разрешении) (not at this resolution)	–(2^{23})	0x00800000
+ INFINITY (БЕСКОНЕЧНОСТЬ)	+ (2^{23} – 2)	0x007FFFFE
— INFINITY (БЕСКОНЕЧНОСТЬ)	– (2^{23} – 2)	0x00800002
Зарезервировано для дальнейшего использования	– (2^{23} – 1)	0x00800001

В каждом из этих специальных случаев экспонента будет равна нулю. Это дает следующие диапазоны представления чисел:

- –128 ≤ показатель ≤ 127
- –2 ((2^{23}) – 3) ≤ мантисса ≤ + ((2^{23}) – 3)
- NaN = + ((2^{23}) – 1)
- NRes = – 2(2**23)
- ± БЕСКОНЕЧНОСТЬ = ± ((2^{23}) – 2)

NaN следует применять для индикации неверного результата вычислительного шага или для индикации потерянных данных, связанных с неспособностью аппаратуры предоставить корректные измерения, возможно из-за сбоя датчика. Менеджер должен отразить эту информацию очисткой дисплея или каким-либо другим способом.

NRes следует применять для индексации того, что значение не может быть представлено при доступном диапазоне и разрешении. Такая ситуация может быть последствием переполнения снизу или сверху, когда число требующихся значимых цифр превышает максимальный или минимальный диапазон экспоненты, или же когда число превышает максимальный или минимальный диапазон экспоненты.

Чтобы сохранить непрерывный диапазон специальных значений, значение бита 0x00800001 сохранено для будущего применения.

F.7 Структура коротких данных с плавающей точкой SFLOAT-Type

Тип SFLOAT-Type может использоваться для представления чисел с плавающей точкой с очень ограниченным диапазоном значений, что значительно сокращает размер полезной информации.

SFLOAT-Type это 16-битное слово, содержащее 4-битный целочисленный показатель со знаком, за которым следует 24 битная целочисленная мантисса со знаком. См. рисунок F.10.

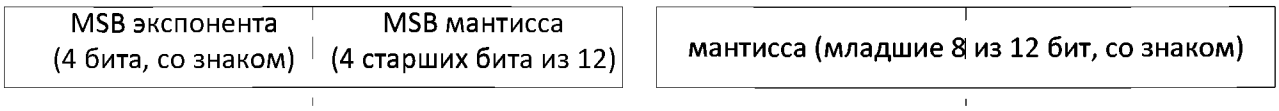


Рисунок F.10 — Короткое представление чисел с плавающей точкой

Представленное число будет иметь вид: (мантисса)×$10^{\text{экспонента}}$. И экспонента и мантисса записываются в двоичном дополнительном коде. Для обозначения точности мантисса и экспонента выравниваются, как описано в F.8.

Существуют специальные значения, которые представлены в таблице F.3.

Таблица F.3 — Специальные значения SFLOAT-Type

Специальное значение	Мантисса	Битовое значение
NaN (не число)	+ (2^{11} – 1)	0x07FF
NRes (не при таком разрешении) (not at this resolution)	–(2^{11})	0x0800
+ INFINITY (БЕСКОНЕЧНОСТЬ)	+ (2^{11} – 2)	0x07FE
– INFINITY (БЕСКОНЕЧНОСТЬ)	–(2^{11} – 2)	0x0802
Зарезервировано для дальнейшего использования	–(2^{11} – 1)	0x0801

Специальные значения выражают следующие значения:

- NaN [экспонента равна 0, мантисса равна + (2^{11} – 1) → 0x07FF];
- NRes [экспонента равна 0, , мантисса равна – (2^{11}) → 0x0800];
- + INFINITY [экспонента равна 0, , мантисса равна + (2^{11} – 2) → 0x07FE];
- – INFINITY [экспонента равна 0, , мантисса равна –(2^{11} – 2) → 0x0802].

В каждом из этих специальных случаев экспонента будет равна нулю. Использование экспоненты для указания допустимых цифр в описании SFLOAT совпадает с F.8.

Чтобы сохранить непрерывность диапазона специальных значений, битовое значение 0x0801 зарезервировано, но оно не представляет определенного числового значения.

F.8 Представление точности чисел с плавающей точкой

Число с плавающей точкой может быть представлено методом, который обозначает точность значения, представляя мантиссу в целочисленной форме, чтобы обозначить число значащих цифр в числе с плавающей точкой и соответствующим образом подогнать экспоненту. Ниже приводятся примеры:

- если экспонента < 0, то целочисленное значение экспоненты показывает число значащих цифр после точки. См. примеры в таблице F.4.

Таблица F.4 — Примеры при экспоненте < 0

Экспонента	Мантисса	Значение
–3	32 000	32.000
–1	320	32.0

- если экспонента ≥ 0, то число значащих цифр после точки равно нулю и мантисса представляет точность значения. См. примеры в таблице F.5.

Таблица F.5 — Примеры при экспоненте ≥ 0

Экспонента	Мантисса	Значение
1	320	3200
2	32	3200

Значения не требующие представления, находящиеся справа от точки, такие как частота пульса, представляются посредством размещения значения в наименее значащем байте мантиссы с экспонентой равной нулю. Например, значение 72 будет представлено как 0x00000048. Процент насыщения кислородом может быть представлен типом SFLOAT, если разместить значение в наименее значащем байте мантиссы. Например, значение 98 будет представлено как 0x0062. В то же время, если доступна более высокая точность (например, показание с точностью до 0.1 единицы измерения), то значение 72.0 будет представлено как 720×10^{-1}, с мантиссой 0x0002D0 и экспонентой 0xFF и в результате значение будет 0xFF0002D0. Для типа SFLOAT значение 98.0 будет представлено как 980×10^{-1}, с мантиссой из 0x3D4 и экспонентой из 0xF и имеет вид 0xF3D4.

Приложение G
(справочное)

Закодированные определения типов данных

В настоящем приложении приведен пример определения заголовков файлов, которые сформированы на языке ASN.1, представленном в приложении А. Настоящее приложение не включает коды, необходимые для преобразования этих структур в двоичных буферах в процессе передачи для машин с прямым или обратным порядком байтов.

```
#ifndef PHD_TYPES
#define PHD_TYPES
/*
Следующие определения типов, возможно, должны быть изменены в зависимости от компилятора и архитектуры машины.
*/
typedef unsigned char intu8;
typedef unsigned short intu16;
typedef unsigned int intu32;

typedef struct Any
{
  intu16 length;
  intu8 value[1];                          /* первый элемент массива */
} Any;

typedef   intu16   OID_Type;
typedef   intu16   PrivateOid;
typedef   intu16   HANDLE;
typedef   intu16   InstNumber;
typedef   intu16   NomPartition;
#define           NOM_PART_UNSPEC          0
#define           NOM_PART_OBJ             1
#define           NOM_PART_METRIC          2
#define           NOM_PART_ALERT           3
#define           NOM_PART_DIM             4
#define           NOM_PART_VATTR           5
#define           NOM_PART_PGRP            6
#define           NOM_PART_SITES           7
#define           NOM_PART_INFRASTRUCT     8
#define           NOM_PART_FEF             9
#define           NOM_PART_ECG_EXTN        10
#define           NOM_PART_PHD_DM          128
#define           NOM_PART_PHD_HF          129
#define           NOM_PART_PHD_AI          130
#define           NOM_PART_RET_CODE        255
#define           NOM_PART_EXT_NOM         256
#define           NOM_PART_PRIV            1024

typedef struct TYPE
{
  NomPartition partition;
  OID_Type code;
} TYPE;

typedef struct AVA_Type
{
  OID_Type attribute_id;
  Any attribute_value;
} AVA_Type;
```

```
typedef struct AttributeList
{
  intu16 count;
  intu16 length;
  AVA_Type value[1];                    /* первый элемент массива */
} AttributeList;

typedef struct AttributeIdList
{
  intu16 count;
  intu16 length;
  OID_Type value[1];                     /* первый элемент массива */
} AttributeIdList;

typedef  intu32 FLOAT_Type;

typedef  intu16 SFLOAT_Type;

typedef  intu32 RelativeTime;

typedef  struct HighResRelativeTime
{
  intu8    value[8];
} HighResRelativeTime;

typedef  struct AbsoluteTimeAdjust
{
  intu8    value[6];
} AbsoluteTimeAdjust;

typedef struct AbsoluteTime
{
  intu8 century;
  intu8 year;
  intu8 month;
  intu8 day;
  intu8 hour;
  intu8 minute;
  intu8 second;
  intu8 sec_fractions;
} AbsoluteTime;

typedef intu16 OperationalState;
#define         OS_DISABLED             0
#define         OS_ENABLED              1
#define         OS_NOT_AVAILABLE        2
typedef struct octet_string
{
  intu16 length;
  intu8 value[1];                       /* первый элемент массива */
} octet_string;

typedef struct SystemModel
{
  octet_string manufacturer;
  octet_string model_number;
} SystemModel;

typedef struct ProdSpecEntry
{
intu16 spec_type;
#define         UNSPECIFIED             0
#define         SERIAL_NUMBER           1
#define         PART_NUMBER             2
#define         HW_REVISION             3
#define         SW_REVISION             4
#define         FW_REVISION             5
```

```
#define      PROTOCOL_REVISION              6
#define      PROD_SPEC_GMDN                 7
PrivateOid component_id;
octet_string prod_spec;
} ProdSpecEntry;

typedef struct ProductionSpec
{
intu16 count;
intu16 length;
ProdSpecEntry value[1];                     /* первый элемент массива */
} ProductionSpec;

typedef intu16 PowerStatus;
#define      ON_MAINS                       0x8000
#define      ON_BATTERY                     0x4000
#define      CHARGING_FULL                  0x0080
#define      CHARGING_TRICKLE               0x0040
#define      CHARGING_OFF                   0x0020

typedef struct BatMeasure
{
FLOAT_Type value;
OID_Type unit;
} BatMeasure;

typedef intu16 MeasurementStatus;
#define      MS_INVALID                     0x8000
#define      MS_QUESTIONABLE                0x4000
#define      MS_NOT_AVAILABLE               0x2000
#define      MS_CALIBRATION_ONGOING         0x1000
#define      MS_TEST_DATA                   0x0800
#define      MS_DEMO_DATA                   0x0400
#define      MS_VALIDATED_DATA              0x0080
#define      MS_EARLY_INDICATION            0x0040
#define      MS_MSMT_ONGOING                0x0020

typedef struct NuObsValue
{
  OID_Type metric_id;
  MeasurementStatus state;
  OID_Type unit_code;
  FLOAT_Type value;
} NuObsValue;

typedef struct NuObsValueCmp
{
  intu16 count;
  intu16 length;
  NuObsValue value[1];                      /* первый элемент массива */
} NuObsValueCmp;

typedef struct SampleType
{
  intu8 sample_size;
  intu8 significant_bits;
} SampleType;
#define SAMPLE_TYPE_SIGNIFICANT_BITS_SIGNED_SAMPLES 255

typedef intu16 SaFlags;
#define      SMOOTH_CURVE                   0x8000
#define      DELAYED_CURVE                  0x4000
#define      STATIC_SCALE                   0x2000
#define      SA_EXT_VAL_RANGE               0x1000
```

```
typedef struct SaSpec
{
  intu16 array_size;
  SampleType sample_type;
  SaFlags flags;
} SaSpec;

typedef struct ScaleRangeSpec8
{
  FLOAT_Type lower_absolute_value;
  FLOAT_Type upper_absolute_value;
  intu8 lower_scaled_value;
  intu8 upper_scaled_value;
} ScaleRangeSpec8;

typedef struct ScaleRangeSpec16
{
  FLOAT_Type lower_absolute_value;
  FLOAT_Type upper_absolute_value;
  intu16 lower_scaled_value;
  intu16 upper_scaled_value;
} ScaleRangeSpec16;

typedef struct ScaleRangeSpec32
{
  FLOAT_Type lower_absolute_value;
  FLOAT_Type upper_absolute_value;
  intu32 lower_scaled_value;
  intu32 upper_scaled_value;
} ScaleRangeSpec32;

typedef struct EnumVal
{
intu16 choice;
intu16 length;
#define          OBJ_ID_CHOSEN              0x0001
#define          TEXT_STRING_CHOSEN         0x0002
#define          BIT_STR_CHOSEN             0x0010

union
{
    OID_Type enum_obj_id;
    octet_string enum_text_string;
    intu32 enum_bit_str; // BITS-32
  } u;
} EnumVal;

typedef struct EnumObsValue
{
  OID_Type metric_id;
  MeasurementStatus state;
  EnumVal value;
} EnumObsValue;

typedef struct AttrValMapEntry
  {
  OID_Type attribute_id;
  intu16 attribute_len;
} AttrValMapEntry;

typedef struct AttrValMap
{
  intu16 count;
  intu16 length;
  AttrValMapEntry value[1];                  /* первый элемент массива */
} AttrValMap;
```

```
typedef struct HandleAttrValMapEntry
{
  HANDLE obj_handle;
  AttrValMap attr_val_map;
} HandleAttrValMapEntry;

typedef intu16 ConfirmMode;
#define        UNCONFIRMED                    0x0000
#define        CONFIRMED                      0x0001

typedef struct HandleAttrValMap
{
  intu16 count;
  intu16 length;
  HandleAttrValMapEntry value[1];             /* первый элемент массива */
} HandleAttrValMap;

typedef intu16 StoSampleAlg;
#define        ST_ALG_NOS                     0x0000
#define        ST_ALG_MOVING_AVERAGE          0x0001
#define        ST_ALG_RECURSIVE_              0x0002
#define        ST_ALG_MIN_PICK                0x0003
#define        ST_ALG_MAX_PICK                0x0004
#define        ST_ALG_MEDIAN                  0x0005
#define        ST_ALG_TRENDED                 0x0200
#define        ST_ALG_NO_DOWNSAMPLING         0x0400

typedef struct SetTimeInvoke
{
  AbsoluteTime date_time;
  FLOAT_Type accuracy;
} SetTimeInvoke;

typedef struct SegmIdList
{
  intu16 count;
  intu16 length;
  InstNumber value[1];                        /* первый элемент массива */
} SegmIdList;

typedef struct AbsTimeRange
{
  AbsoluteTime from_time;
  AbsoluteTime to_time;
} AbsTimeRange;

typedef struct SegmentInfo
{
  InstNumber seg_inst_no;
  AttributeList seg_info;
} SegmentInfo;

typedef struct SegmentInfoList
{
  intu16 count;
  intu16 length;
  SegmentInfo value[1];                       /* первый элемент массива */
} SegmentInfoList;

typedef struct SegmSelection
{
    intu16 choice;
    intu16 length;
#define            ALL_SEGMENTS_CHOSEN              0x0001
#define            SEGM_ID_LIST_CHOSEN              0x0002
#define            ABS_TIME_RANGE_CHOSEN            0x0003
   union
    {
```

```
    intu16 all_segments;
    SegmIdList segm_id_list;
    AbsTimeRange abs_time_range;
  } u;
} SegmSelection;

typedef intu16 PMStoreCapab;
#define       PMSC_VAR_NO_OF_SEGM                    0x8000
#define       PMSC_EPI_SEG_ENTRIES                   0x0800
#define       PMSC_PERI_SEG_ENTRIES                  0x0400
#define       PMSC_ABS_TIME_SELECT                   0x0200
#define       PMSC_CLEAR_SEGM_BY_LIST_SUP            0x0100
#define       PMSC_CLEAR_SEGM_BY_TIME_SUP            0x0080
#define       PMSC_CLEAR_SEGM_REMOVE                 0x0040
#define       PMSC_MULTI_PERSON                      0x0008

typedef intu16 SegmEntryHeader;
#define       SEG_ELEM_HDR_ABSOLUTE_TIME             0x8000
#define       SEG_ELEM_HDR_RELATIVE_TIME             0x4000
#define       SEG_ELEM_HDR_HIRES_RELATIVE_TIME       0x2000

typedef struct SegmEntryElem
{
  OID_Type class_id;
  TYPE metric_type;
  HANDLE handle;
  AttrValMap attr_val_map;
} SegmEntryElem;

typedef struct SegmEntryElemList
{
  intu16 count;
  intu16 length;
  SegmEntryElem value[1];                             /* первый элемент массива */
} SegmEntryElemList;

typedef struct PmSegmentEntryMap
{
  SegmEntryHeader segm_entry_header;
  SegmEntryElemList segm_entry_elem_list;
} PmSegmentEntryMap;

typedef struct SegmElemStaticAttrEntry
{
  OID_Type class_id;
  TYPE metric_type;
  AttributeList attribute_list;
} SegmElemStaticAttrEntry;

typedef struct PmSegmElemStaticAttrList
{
  intu16 count;
  intu16 length;
  SegmElemStaticAttrEntry value[1];                   /* первый элемент массива */
} PmSegmElemStaticAttrList;

typedef struct TrigSegmDataXferReq
{
  InstNumber seg_inst_no;
} TrigSegmDataXferReq;

typedef intu16 TrigSegmXferRsp;
#define       TSXR_SUCCESSFUL                        0
#define       TSXR_FAIL_NO_SUCH_SEGMENT              1
#define       TSXR_FAIL_SEGM_TRY_LATER               2
#define       TSXR_FAIL_SEGM_EMPTY                   3
#define       TSXR_FAIL_OTHER                        512
```

```
typedef struct TrigSegmDataXferRsp
{
  InstNumber seg_inst_no;
  TrigSegmXferRsp trig_segm_xfer_rsp;
} TrigSegmDataXferRsp;

typedef intu16 SegmEvtStatus;
#define      SEVTSTA_FIRST_ENTRY              0x8000
#define      SEVTSTA_LAST_ENTRY               0x4000
#define      SEVTSTA_AGENT_ABORT              0x0800
#define      SEVTSTA_MANAGER_CONFIRM          0x0080
#define      SEVTSTA_MANAGER_ABORT            0x0008

typedef struct SegmDataEventDescr
{
  InstNumber segm_instance;
  intu32 segm_evt_entry_index;
  intu32 segm_evt_entry_count;
  SegmEvtStatus segm_evt_status;
} SegmDataEventDescr;

typedef struct SegmentDataEvent
{
  SegmDataEventDescr segm_data_event_descr;
  octet_string segm_data_event_entries;
} SegmentDataEvent;

typedef struct SegmentDataResult
{
  SegmDataEventDescr segm_data_event_descr;
} SegmentDataResult;

typedef intu16 SegmStatType;
#define      SEGM_STAT_TYPE_MINIMUM           1
#define      SEGM_STAT_TYPE_MAXIMUM           2
#define      SEGM_STAT_TYPE_AVERAGE           3

typedef struct SegmentStatisticEntry
{
  SegmStatType segm_stat_type;
  octet_string segm_stat_entry;
} SegmentStatisticEntry;

typedef struct SegmentStatistics
{
  intu16 count;
  intu16 length;
  SegmentStatisticEntry value[1];              /* первый элемент массива */
} SegmentStatistics;

typedef struct ObservationScan
{
  HANDLE obj_handle;
  AttributeList attributes;
} ObservationScan;

typedef OID_Type TimeProtocolId;

typedef intu32 AssociationVersion;
#define      ASSOC_VERSION1                   0x80000000

typedef intu32 ProtocolVersion;
#define      PROTOCOL_VERSION1                0x80000000

typedef intu16 EncodingRules;
#define      MDER                             0x8000
#define      XER                              0x4000
#define      PER                              0x2000
```

```
typedef struct UUID_Ident
{
  intu8 value[16];
} UUID_Ident;

typedef intu16 DataProtoId;
#define        DATA_PROTO_ID_20601                          20601
#define        DATA_PROTO_ID_EXTERNAL                       65535

typedef struct DataProto
{
  DataProtoId data_proto_id;
  Any data_proto_info;
} DataProto;

typedef struct DataProtoList
{
  intu16 count;
  intu16 length;
  DataProto value[1];                                       /* первый элемент массива */
} DataProtoList;

typedef struct AARQ_apdu
{
  AssociationVersion assoc_version;
  DataProtoList data_proto_list;
} AARQ_apdu;

typedef intu16 Associate_result;
#define        ACCEPTED                                     0
#define        REJECTED_PERMANENT                           1
#define        REJECTED_TRANSIENT                           2
#define        ACCEPTED_UNKNOWN_CONFIG                      3
#define        REJECTED_NO_COMMON_PROTOCOL                  4
#define        REJECTED_NO_COMMON_PARAMETER                 5
#define        REJECTED_UNKNOWN                             6
#define        REJECTED_UNAUTHORIZED                        7
#define        REJECTED_UNSUPPORTED_ASSOC_VERSION           8

typedef struct AARE_apdu
{
  Associate_result result;
  DataProto selected_data_proto;
} AARE_apdu;

typedef intu16 Release_request_reason;
#define        RELEASE_REQUEST_REASON_NORMAL                0

typedef struct RLRQ_apdu
{
  Release_request_reason reason;
} RLRQ_apdu;

typedef intu16 Release_response_reason;
#define        RELEASE_RESPONSE_REASON_NORMAL               0
typedef struct RLRE_apdu
{
Release_response_reason reason;
} RLRE_apdu;

typedef intu16 Abort_reason;
#define        ABORT_REASON_UNDEFINED                       0
#define        ABORT_REASON_BUFFER_OVERFLOW                 1
#define        ABORT_REASON_RESPONSE_TIMEOUT                2
#define        ABORT_REASON_CONFIGURATION_TIMEOUT           3

typedef struct ABRT_apdu
{
```

```
  Abort_reason reason;
} ABRT_apdu;

typedef octet_string PRST_apdu;

typedef intu16 InvokeIDType;

typedef struct EventReportArgumentSimple
{
  HANDLE obj_handle;
  RelativeTime event_time;
  OID_Type event_type;
  Any event_info;
} EventReportArgumentSimple;

typedef struct GetArgumentSimple
{
  HANDLE obj_handle;
  AttributeIdList attribute_id_list;
} GetArgumentSimple;

typedef intu16 ModifyOperator;
#define       REPLACE                 0
#define       ADD_VALUES              1
#define       REMOVE_VALUES           2
#define       SET_TO_DEFAULT          3

typedef struct AttributeModEntry
{
  ModifyOperator modify_operator;
  AVA_Type attribute;
} AttributeModEntry;

typedef struct ModificationList
{
  intu16 count;
  intu16 length;
  AttributeModEntry value[1];           /* первый элемент массива */
} ModificationList;

typedef struct SetArgumentSimple
{
  HANDLE obj_handle;
  ModificationList modification_list;
} SetArgumentSimple;

typedef struct ActionArgumentSimple
{
  HANDLE obj_handle;
  OID_Type action_type;
  Any action_info_args;
} ActionArgumentSimple;

typedef struct EventReportResultSimple
{
  HANDLE obj_handle;
  RelativeTime currentTime;
  OID_Type event_type;
  Any event_reply_info;
} EventReportResultSimple;

typedef struct GetResultSimple
{
  HANDLE obj_handle;
  AttributeList attribute_list;
} GetResultSimple;
```

```
typedef struct TypeVer
{
  OID_Type type;
  intu16 version;
} TypeVer;

typedef struct TypeVerList
{
  intu16 count;
  intu16 length;
  TypeVer value[1];                                   /* первый элемент массива */
} TypeVerList;

typedef struct SetResultSimple
{
  HANDLE obj_handle;
  AttributeList attribute_list;
} SetResultSimple;

typedef struct ActionResultSimple
{
  HANDLE obj_handle;
  OID_Type action_type;
  Any action_info_args;
} ActionResultSimple;

typedef intu16 ERROR;
#define       NO_SUCH_OBJECT_INSTANCE          1
#define       ACCESS_DENIED                    2
#define       NO_SUCH_ACTION                   9
#define       INVALID_OBJECT_INSTANCE          17
#define       PROTOCOL_VIOLATION               23
#define       NOT_ALLOWED_BY_OBJECT            24
#define       ACTION_TIMED_OUT                 25
#define       ACTION_ABORTED                   26

typedef struct ErrorResult
{
  ERROR error_value;
  Any parameter;
} ErrorResult;

typedef intu16 RorjProblem;
#define       UNRECOGNIZED_APDU                0
#define       BADLY_STRUCTURED_APDU            2
#define       UNRECOGNIZED_OPERATION           101
#define       RESOURCE_LIMITATION              103
#define       UNEXPECTED_ERROR                 303

typedef struct RejectResult
{
  RorjProblem problem;
} RejectResult;

typedef struct DATA_apdu
{
  InvokeIDType invoke_id;
  struct
{
intu16 choice;
intu16 length;
#define       ROIV_CMIP_EVENT_REPORT_CHOSEN                    0x0100
#define       ROIV_CMIP_CONFIRMED_EVENT_REPORT_CHOSEN          0x0101
#define       ROIV_CMIP_GET_CHOSEN                             0x0103
#define       ROIV_CMIP_SET_CHOSEN                             0x0104
#define       ROIV_CMIP_CONFIRMED_SET_CHOSEN                   0x0105
```

```
#define     ROIV_CMIP_ACTION_CHOSEN                          0x0106
#define     ROIV_CMIP_CONFIRMED_ACTION_CHOSEN                0x0107
#define     RORS_CMIP_CONFIRMED_EVENT_REPORT_CHOSEN          0x0201
#define     RORS_CMIP_GET_CHOSEN                             0x0203
#define     RORS_CMIP_CONFIRMED_SET_CHOSEN                   0x0205
#define     RORS_CMIP_CONFIRMED_ACTION_CHOSEN                0x0207
#define     ROER_CHOSEN                                      0x0300
#define     RORJ_CHOSEN                                      0x0400

    union
    {
      EventReportArgumentSimple roiv_cmipEventReport;
      EventReportArgumentSimple roiv_cmipConfirmedEventReport;
      GetArgumentSimple roiv_cmipGet;
      SetArgumentSimple roiv_cmipSet;
      SetArgumentSimple roiv_cmipConfirmedSet;
      ActionArgumentSimple roiv_cmipAction;
      ActionArgumentSimple roiv_cmipConfirmedAction;
      EventReportResultSimple rors_cmipConfirmedEventReport;
      GetResultSimple rors_cmipGet;
      SetResultSimple rors_cmipConfirmedSet;
      ActionResultSimple rors_cmipConfirmedAction;
      ErrorResult roer;
      RejectResult rorj;
    } u;
  } choice;
} DATA_apdu;

typedef struct APDU
{
    intu16 choice;
    intu16 length;
#define     AARQ_CHOSEN                  0xE200
#define     AARE_CHOSEN                  0xE300
#define     RLRQ_CHOSEN                  0xE400
#define     RLRE_CHOSEN                  0xE500
#define     ABRT_CHOSEN                  0xE600
#define     PRST_CHOSEN                  0xE700

  union
  {
      AARQ_apdu aarq;
      AARE_apdu aare;
      RLRQ_apdu rlrq;
      RLRE_apdu rlre;
      ABRT_apdu abrt;
      PRST_apdu prst;
    } u;
} APDU;

typedef intu32 NomenclatureVersion;
#define     NOM_VERSION1                     0x80000000

typedef intu32 FunctionalUnits;
#define     FUN_UNITS_UNIDIRECTIONAL         0x80000000
#define     FUN_UNITS_HAVETESTCAP            0x40000000
#define     FUN_UNITS_CREATETESTASSOC        0x20000000

typedef intu32 SystemType;
#define     SYS_TYPE_MANAGER                 0x80000000
#define     SYS_TYPE_AGENT                   0x00800000

typedef intu16 ConfigId;
#define     MANAGER_CONFIG_RESPONSE          0x0000
#define     STANDARD_CONFIG_START            0x0001
#define     STANDARD_CONFIG_END              0x3FFF
```

```
#define     EXTENDED_CONFIG_START                   0x4000
#define     EXTENDED_CONFIG_END                     0x7FFF
#define     RESERVED_START                          0x8000
#define     RESERVED_END                            0xFFFF

typedef struct DataReqModeCapab
{
  DataReqModeFlags data_req_mode_flags;
  intu8 data_req_init_agent_count;
  intu8 data_req_init_manager_count;
} DataReqModeCapab;

typedef intu16 DataReqModeFlags;
#define     DATA_REQ_SUPP_STOP                      0x8000
#define     DATA_REQ_SUPP_SCOPE_ALL                 0x0800
#define     DATA_REQ_SUPP_SCOPE_CLASS               0x0400
#define     DATA_REQ_SUPP_SCOPE_HANDLE              0x0200
#define     DATA_REQ_SUPP_MODE_SINGLE_RSP           0x0080
#define     DATA_REQ_SUPP_MODE_TIME_PERIOD          0x0040
#define     DATA_REQ_SUPP_MODE_TIME_NO_LIMIT        0x0020
#define     DATA_REQ_SUPP_PERSON_ID                 0x0010
#define     DATA_REQ_SUPP_INIT_AGENT                0x0001

typedef struct PhdAssociationInformation
{
  ProtocolVersion protocolVersion;
  EncodingRules encodingRules;
  NomenclatureVersion nomenclatureVersion;
  FunctionalUnits functionalUnits;
  SystemType systemType;
  octet_string system_id;
  intu16 dev_config_id;
  DataReqModeCapab data_req_mode_capab;
  AttributeList optionList;
} PhdAssociationInformation;

typedef struct ManufSpecAssociationInformation
{
  UUID_Ident data_proto_id_ext;
  Any data_proto_info_ext;
} ManufSpecAssociationInformation;

typedef intu16 MdsTimeCapState;
#define     MDS_TIME_CAPAB_REAL_TIME_CLOCK          0x8000
#define     MDS_TIME_CAPAB_SET_CLOCK                0x4000
#define     MDS_TIME_CAPAB_RELATIVE_TIME            0x2000
#define     MDS_TIME_CAPAB_HIGH_RES_RELATIVE_TIME   0x1000
#define     MDS_TIME_CAPAB_SYNC_ABS_TIME            0x0800
#define     MDS_TIME_CAPAB_SYNC_REL_TIME            0x0400
#define     MDS_TIME_CAPAB_SYNC_HI_RES_RELATIVE_TIME 0x0200
#define     MDS_TIME_STATE_ABS_TIME_SYNCED          0x0080
#define     MDS_TIME_STATE_REL_TIME_SYNCED          0x0040
#define     MDS_TIME_STATE_HI_RES_RELATIVE_TIME_SYNCED 0x0020
#define     MDS_TIME_MGR_SET_TIME                   0x0010

typedef struct MdsTimeInfo
{
  MdsTimeCapState mds_time_cap_state;
  TimeProtocolId time_sync_protocol;
  RelativeTime time_sync_accuracy;
  intu16 time_resolution_abs_time;
  intu16 time_resolution_rel_time;
  intu32 time_resolution_high_res_time;
} MdsTimeInfo;
```

```
typedef octet_string EnumPrintableString;

typedef intu16 PersonId;
#define     UNKNOWN_PERSON_ID                   0xFFFF

typedef intu16 MetricSpecSmall;
#define     MSS_AVAIL_INTERMITTENT              0x8000
#define     MSS_AVAIL_STORED_DATA               0x4000
#define     MSS_UPD_APERIODIC                   0x2000
#define     MSS_MSMT_APERIODIC                  0x1000
#define     MSS_MSMT_PHYS_EV_ID                 0x0800
#define     MSS_MSMT_BTB_METRIC                 0x0400
#define     MSS_ACC_MANAGER_INITIATED           0x0080
#define     MSS_ACC_AGENT_INITIATED             0x0040
#define     MSS_CAT_MANUAL                      0x0008
#define     MSS_CAT_SETTING                     0x0004
#define     MSS_CAT_CALCULATION                 0x0002

typedef struct MetricStructureSmall
{
    intu8 ms-struct;
#define MS_STRUCT_SIMPLE 0
#define MS_STRUCT_COMPOUND 1
#define MS_STRUCT_RESERVED 2
#define MS_STRUCT_COMPOUND_FIX 3
    intu8 ms-comp-no;
} MetricStructureSmall;

typedef struct MetricIdList
{
  intu16 count;
  intu16 length;
  OID-Type value[1];                            /* первый элемент массива */
} MetricIdList;

typedef struct SupplementalTypeList
{
  intu16 count;
  intu16 length;
  TYPE value[1];                                /* первый элемент массива */
} SupplementalTypeList;

typedef struct ObservationScanList
{
  intu16 count;
  intu16 length;
  ObservationScan value[1];                     /* первый элемент массива */
} ObservationScanList;

typedef struct ScanReportPerVar
{
  intu16 person_id;
  ObservationScanList obs_scan_var;
} ScanReportPerVar;

typedef struct ScanReportPerVarList
{
  intu16 count;
  intu16 length;
  ScanReportPerVar value[1];                    /* первый элемент массива */
} ScanReportPerVarList;

typedef intu16 DataReqId;
#define     DATA_REQ_ID_MANAGER_INITIATED_MIN       0x0000
#define     DATA_REQ_ID_MANAGER_INITIATED_MAX       0xEFFF
#define     DATA_REQ_ID_AGENT_INITIATED             0xF000
```

```
typedef struct ScanReportInfoMPVar
{
  DataReqId data_req_id;
  intu16 scan_report_no;
  ScanReportPerVarList scan_per_var;
} ScanReportInfoMPVar;

typedef struct ObservationScanFixed
{
  HANDLE obj_handle;
  octet_string obs_val_data;
} ObservationScanFixed;

typedef struct ObservationScanFixedList
{
  intu16 count;
  intu16 length;
  ObservationScanFixed value[1];          /* первый элемент массива */
} ObservationScanFixedList;

typedef struct ScanReportPerFixed
{
  intu16 person_id;
  ObservationScanFixedList obs_scan_fix;
} ScanReportPerFixed;

typedef struct ScanReportPerFixedList
{
  intu16 count;
  intu16 length;
  ScanReportPerFixed value[1];            /* первый элемент массива */
} ScanReportPerFixedList;

typedef struct ScanReportInfoMPFixed
{
  DataReqId data_req_id;
  intu16 scan_report_no;
  ScanReportPerFixedList scan_per_fixed;
} ScanReportInfoMPFixed;

typedef struct ScanReportInfoVar
{
  DataReqId data_req_id;
  intu16 scan_report_no;
  ObservationScanList obs_scan_var;
} ScanReportInfoVar;

typedef struct ScanReportInfoFixed
{
  DataReqId data_req_id;
  intu16 scan_report_no;
  ObservationScanFixedList obs_scan_fixed;
} ScanReportInfoFixed;

typedef octet_string ObservationScanGrouped;

typedef struct ScanReportInfoGroupedList
{
  intu16 count;
  intu16 length;
  ObservationScanGrouped value[1];        /* первый элемент массива */
} ScanReportInfoGroupedList;

typedef struct ScanReportInfoGrouped
{
  intu16 data_req_id;
  intu16 scan_report_no;
```

```
  ScanReportInfoGroupedList obs_scan_grouped;
} ScanReportInfoGrouped;

typedef struct ScanReportPerGrouped
{
  PersonId person_id;
  ObservationScanGrouped obs_scan_grouped;
} ScanReportPerGrouped;

typedef struct ScanReportPerGroupedList
{
  intu16 count;
  intu16 length;
  ScanReportPerGrouped value[1];          /* первый элемент массива */
} ScanReportPerGroupedList;

typedef struct ScanReportInfoMPGrouped
{
  intu16 data_req_id;
  intu16 scan_report_no;
  ScanReportPerGroupedList scan_per_grouped;
} ScanReportInfoMPgrouped;

typedef struct ConfigObject
{
  OID_Type obj_class;
  HANDLE obj_handle;
  AttributeList attributes;
} ConfigObject;

typedef struct ConfigObjectList
{
  intu16 count;
  intu16 length;
  ConfigObject value[1];                  /* первый элемент массива */
} ConfigObjectList;

typedef struct ConfigReport
{
  ConfigId config_report_id;
  ConfigObjectList config_obj_list;
} ConfigReport;

typedef intu16 ConfigResult;
#define     ACCEPTED_CONFIG                 0x0000
#define     UNSUPPORTED_CONFIG              0x0001
#define     STANDARD_CONFIG_UNKNOWN         0x0002

typedef struct ConfigReportRsp
{
  ConfigId config_report_id;
  ConfigResult config_result;
} ConfigReportRsp;

typedef intu16 DataReqMode;
#define     DATA_REQ_START_STOP                 0x8000
#define     DATA_REQ_CONTINUATION               0x4000
#define     DATA_REQ_SCOPE_ALL                  0x0800
#define     DATA_REQ_SCOPE_TYPE                 0x0400
#define     DATA_REQ_SCOPE_HANDLE               0x0200
#define     DATA_REQ_MODE_SINGLE_RSP            0x0080
#define     DATA_REQ_MODE_TIME_PERIOD           0x0040
#define     DATA_REQ_MODE_TIME_NO_LIMIT         0x0020
#define     DATA_REQ_MODE_DATA_REQ_PERSON_ID    0x0008
```

```
typedef struct HANDLEList
{
  intu16 count;
  intu16 length;
  HANDLE value[1];                                  /* первый элемент массива */
} HANDLEList;

typedef struct DataRequest
{
  DataReqId data_req_id;
  DataReqMode data_req_mode;
  RelativeTime data_req_time;
  intu16 DataRequest_data_req_person_id;
  OID_Type data_req_class;
  HANDLEList data_req_obj_handle_list;
} DataRequest;

typedef intu16 DataReqResult;
#define     DATA_REQ_RESULT_NO_ERROR                          0
#define     DATA_REQ_RESULT_UNSPECIFIC_ERROR                  1
#define     DATA_REQ_RESULT_NO_STOP_SUPPORT                   2
#define     DATA_REQ_RESULT_NO_SCOPE_ALL_SUPPORT              3
#define     DATA_REQ_RESULT_NO_SCOPE_CLASS_SUPPORT            4
#define     DATA_REQ_RESULT_NO_SCOPE_HANDLE_SUPPORT           5
#define     DATA_REQ_RESULT_NO_MODE_SINGLE_RSP_SUPPORT        6
#define     DATA_REQ_RESULT_NO_MODE_TIME_PERIOD_SUPPORT       7
#define     DATA_REQ_RESULT_NO_MODE_TIME_NO_LIMIT_SUPPORT     8
#define     DATA_REQ_RESULT_NO_PERSON_ID_SUPPORT              9
#define     DATA_REQ_RESULT_UNKNOWN_PERSON_ID                 11
#define     DATA_REQ_RESULT_UNKNOWN_CLASS                     12
#define     DATA_REQ_RESULT_UNKNOWN_HANDLE                    13
#define     DATA_REQ_RESULT_UNSUPP_SCOPE                      14
#define     DATA_REQ_RESULT_UNSUPP_MODE                       15
#define     DATA_REQ_RESULT_INIT_MANAGER_OVERFLOW             16
#define     DATA_REQ_RESULT_CONTINUATION_NOT_SUPPORTED        17
#define     DATA_REQ_RESULT_INVALID_REQ_ID                    18

typedef struct DataResponse
{
  RelativeTime rel_time_stamp;
  DataReqResult data_req_result;
  OID_Type event_type;
  Any event_info;
} DataResponse;

typedef FLOAT_Type SimpleNuObsValue;

typedef struct SimpleNuObsValueCmp
{
  intu16 count;
  intu16 length;
  SimpleNuObsValue value[1];                    /* первый элемент массива */
} SimpleNuObsValueCmp;

typedef SFLOAT_Type BasicNuObsValue;

typedef struct BasicNuObsValueCmp
{
  intu16 count;
  intu16 length;
  BasicNuObsValue value[1];                     /* первый элемент массива */
} BasicNuObsValueCmp;
#endif /* PHD_TYPES */
```

Приложение Н
(справочное)

Примеры

Н.1 Общие положения

Настоящее приложение представляет двоичные примеры сообщений, которыми обмениваются агент и менеджер.

Н.2 Взвешенная шкала

Н.2.1 Ассоциация

Н.2.1.1 Запрос ассоциации

Агент отправляет менеджеру следующее сообщение. Этот пример предполагает, что агент описывается набором взвешенных шкал.

```
0xE2  0x00                    APDU CHOICE Type (AarqApdu)
0x00  0x32                    CHOICE.length = 50
0x80  0x00  0x00  0x00        assoc-version
0x00  0x01  0x00  0x2A              data-proto-list.count = 1 | length = 42
0x50  0x79                    data-proto-id = 20601
0x00  0x26                    data-proto-info length = 38
0x80  0x00  0x00  0x00        protocolVersion
0xA0  0x00                    encoding rules = MDER or PER
0x80  0x00  0x00  0x00        nomenclatureVersion
0x00  0x00  0x00  0x00        functionalUnits — например, пометить возможность отправления отчетов
                              о незапрашиваемых событиях
0x00  0x80  0x00  0x00        systemType = sys-type-agent
0x00  0x08                    system-id length = 8 and value
0x88  0x77  0x66  0x55  0x44  0x33  0x22  0x11
0x40  0x00                    dev-config-id
0x00  0x81  0x01 0x01         data-req-mode-capab
0x00  0x00  0x00 0x00         optionList.count = 0 | optionList.length = 0
```

Н.2.1.2 Ответ ассоциации

Менеджер отвечает агенту, что он может вступить в ассоциацию, но необходима конфигурация агента.

```
0xE3  0x00                    APDU CHOICE Type (AareApdu)
0x00  0x2C                    CHOICE.length = 44
0x00  0x03                    result = accepted-unknown-config
0x50  0x79                    data-proto-id = 20601
0x00  0x26  0x00  0x00        protocolVersion
0x80  0x00                    encoding rules = MDER
0x80  0x00  0x00  0x00        nomenclatureVersion
0x00  0x00  0x00  0x00        functionalUnits
0x80  0x00  0x00  0x00        systemType = sys-type-manager
0x00  0x08                    system-id length = 8 and value
0x11  0x22  0x33  0x44  0x55  0x66  0x77  0x88
0x00  0x00                    Ответ менеджера config-id - всегда 0
0x00  0x00  0x00  0x00        Ответ менеджера data-req-mode-capab - всегда 0
0x00  0x00  0x00  0x00        optionList.count = 0 | optionList.length = 0
```

Н.2.2 Обмен информацией о конфигурации

Н.2.2.1 Удаленный вызов отчета о событии конфигурации

Агент сообщает менеджеру свою конфигурацию.

```
0xE7  0x00                    APDU CHOICE Type (PrstApdu)
0x00  0x70                    CHOICE.length = 112
0x00  0x6E                    OCTET STRING.length = 110
0x43  0x21                    invoke-id =   0x4321 (start of DataApdu. MDER encoded.)
0x01  0x01                    CHOICE(Remote Operation Invoke | Confirmed Event Report)
0x00  0x68                    CHOICE.length = 104
0x00  0x00                    obj-handle =   0 (MDS object)
0x00  0x00  0x00  0x00        event-time =   0
```

0x0D	0x1C			event-type = MDC_NOTI_CONFIG
0x00	0x5E			event-info.length = 94 (start of ConfigReport)
0x40	0x00			config-report-id
0x00	0x02			config-obj-list.count = 2 Measurement objects will be “announced”
0x00	0x58			config-obj-list.length = 88
0x00	0x06			obj-class = MDC_MOC_VMO_METRIC_NU
0x00	0x01			obj-handle = 1 (→1^{oe} измерение масса тела)
0x00	0x04			attributes.count =4
0x00	0x24			attributes.length = 36
0x09	0x2F			attribute-id = MDC_ATTR_ID_TYPE
0x00	0x04			attribute-value.length = 4
0x00	0x02	0xE1	0x40	MDC_PART_SCADA | MDC_MASS_BODY_ACTUAL
0x0A	0x46			attribute-id = MDC_ATTR_METRIC_SPEC_SMALL
0x00	0x02			attribute-value.length = 2
0xD0	0x40			intermittent, stored data, msmt aperiodic, agent init, measured
0x09	0x96			attribute-id = MDC_ATTR_UNIT_CODE
0x00	0x02			attribute-value.length = 2
0x06	0xC3			MDC_DIM_KILO_G
0x0A	0x55			attribute-id = MDC_ATTR_ATTRIBUTE_VAL_MAP
0x00	0x0C			attribute-value.length = 12
0x00	0x02			AttrValMap.count = 2
0x00	0x08			AttrValMap.length = 8
0x0A	0x56	0x00	0x04	MDC_ATTR_NU_VAL_OBS_SIMP | value length = 4
0x09	0x90	0x00	0x08	MDC_ATTR_TIME_STAMP_ABS | value length = 8
0x00	0x06			obj-class = MDC_MOC_VMO_METRIC_NU
0x00	0x02			obj-handle = 2 (→2^{oe} измерение масса тела)
0x00	0x04			attributes.count = 4
0x00	0x24			attributes.length = 36
0x09	0x2F			attribute-id = MDC_ATTR_ID_TYPE
0x00	0x04			attribute-value.length = 4
0x00	0x02	0xE1	0x4C	MDC_PART_SCADA | MDC_BODY_FAT
0x0A	0x46			attribute-id = MDC_ATTR_METRIC_SPEC_SMALL
0x00	0x02			attribute-value.length = 2
0xD0	0x42			intermittent, stored data, msmt aperiodic, agent init, calculated
0x09	0x96			attribute-id = MDC_ATTR_UNIT_CODE
0x00	0x02			attribute-value.length = 2
0x02	0x20			MDC_DIM_PERCENT
0x0A	0x55			attribute-id = MDC_ATTR_ATTRIBUTE_VAL_MAP
0x00	0x0C			attribute-value.length = 12
0x00	0x02			AttrValMap.count = 2
0x00	0x08			AttrValMap.length = 8
0x0A	0x56	0x00	0x04	MDC_ATTR_NU_VAL_OBS_SIMP, 4
0x09	0x90	0x00	0x08	MDC_ATTR_TIME_STAMP_ABS, 8

H.2.2.2 Удаленный ответ с отчетом о событии конфигурации

Менеджер отвечает, что он может задействовать конфигурацию агента.

0xE7	0x00			APDU CHOICE Type (PrstApdu)
0x00	0x16			CHOICE.length = 22
0x00	0x14			OCTET STRING.length = 20
0x43	0x21			invoke-id = 0x4321 (start of DataApdu. MDER encoded.)
0x02	0x01			CHOICE (Remote Operation Response | Confirmed Event Report)
0x00	0x0E			CHOICE.length = 14
0x00	0x00			obj-handle = 0 (MDS object)
0x00	0x00	0x00	0x00	currentTime = 0
0x0D	0x1C			event-type = MDC_NOTI_CONFIG
0x00	0x04			event-reply-info.length = 4
0x40	0x00			ConfigReportRsp.config-report-id = 0x4000
0x00	0x00			ConfigReportRsp.config-result = accepted-config

H.2.3 Служба атрибутов GET MDS

H.2.3.1 Общие положения

Атрибуты GET MDS вызываются в любое время, когда устройство находится в состоянии ассоциации.

H.2.3.2 Получение запроса на все атрибуты MDS

Менеджер запрашивает агента об атрибутах его объекта MDS.

```
0xE7  0x00                          APDU CHOICE Type (PrstApdu)
0x00  0x0E                          CHOICE.length = 14
0x00  0x0C                          OCTET STRING.length = 12
0x34  0x56                          invoke-id =   0x3456 (start of DataApdu. MDER encoded.)
0x01  0x03  CHOICE                  (Вызов Удаленной операции | Get)
0x00  0x06                          CHOICE.length = 6
0x00  0x00                          handle =   0 (MDS object)
0x00  0x00  attribute-id-list.count = 0 (all attributes)
0x00  0x00   attribute-id-list.length = 0
```

H.2.3.3 Получение ответов о всех атрибутах MDS

Агент отвечает менеджеру о всех своих атрибутах. В настоящем примере предполагается, что агент поддерживает AbsoluteTime (абсолютное время) и не поддерживает RelativeTime (относительное время). Кроме того, передаются также некоторые дополнительные поля.

```
0xE7  0x00                          APDU CHOICE Type (PrstApdu)
0x00  0x6E                          CHOICE.length = 110
0x00  0x6C                          OCTET STRING.length = 108
0x34  0x56                          invoke-id =   0x3456 (mirrored from request) (start of DataApdu. MDER encoded.)
0x02  0x03                          CHOICE (Remote Operation Response | Get)
0x00  0x66                          CHOICE.length = 102
0x00  0x00                          handle =   0 (MDS object)
0x00  0x06                          attribute-list.count = 6
0x00  0x60                          attribute-list.length = 96
0x0A  0x5A                          attribute id = MDC_ATTR_SYS_TYPE_SPEC_LIST
0x00  0x08                          attribute-value.length = 8
0x00  0x01                          TypeVerList count = 1
0x00  0x04                          TypeVerList length = 4
0x10  0x0F                          type = MDC_DEV_SPEC_PROFILE_SCALE
0x00  0x01                          version = version 1 of the specialization
0x09  0x28                          attribute-id = MDC_ATTR_ID_MODEL
0x00  0x1A                          attribute-value.length = 26
0x00  0x0A  0x54  0x68              string length = 10 | "TheCompany"
0x65  0x43  0x6F  0x6D
0x70  0x61  0x6E  0x79
0x00  0x0C  0x54  0x68              string length = 12 | "TheScaleABC\0"
0x65  0x53  0x63  0x61
0x6C  0x65  0x41  0x42  0x43  0x00
0x09  0x84                          attribute-id = MDC_ATTR_SYS_ID
0x00  0x0A                          attribute-value.length = 10
0x00  0x08  0x88  0x77  0x66  0x55  0x44  0x33  0x22  0x11 octet string length = 8 | EUI-64
0x0a  0x44                          attribute-id = MDC_ATTR_DEV_CONFIG_ID
0x00  0x02                          attribute-value.length = 2
0x40  0x00                          dev-config-id = 16384 (extended-config-start)
0x09  0x2D                          attribute-id = MDC_ATTR_ID_PROD_SPECN
0x00  0x12                          attribute-value.length = 18
0x00  0x01                          ProductionSpec.count = 1
0x00  0x0E                          ProductionSpec.length = 14
0x00  0x01                          ProdSpecEntry.spec-type = 1 (serial-number)
0x00  0x00                          ProdSpecEntry.component-id =   0
0x00  0x08  0x44  0x45              string length = 8 | prodSpecEntry.prod-spec = "DE124567"
0x31  0x32  0x34  0x35
0x36  0x37
0x09  0x87                          attribute-id =MDC_ATTR_TIME_ABS
0x00  0x08                          attribute-value.length = 8
0x20  0x07  0x02  0x01              Absolute-Time-Stamp = 2007-02-01T12:05:0000
0x12  0x05  0x00  0x00
```

H.2.4 Создание отчетов данных

H.2.4.1 Инициируемая агентом передача данных измерений

Агент отправляет непосредственный отчет о событии менеджеру, содержащий результаты измерений.

```
0xE7  0x00                    APDU CHOICE Type (PrstApdu)
0x00  0x3A                    CHOICE.length = 58
0x00  0x38                    OCTET STRING.length = 56
0x43  0x21                    invoke-id =   0x4321 (start of DataApdu. MDER encoded.)
0x01  0x01                    CHOICE(Remote Operation Invoke | Confirmed Event Report)
0x00  0x32                    CHOICE.length = 50
0x00  0x00                    obj-handle =   0 (MDS object)
0x00  0x00  0x00  0x00        event-time =   0
0x0D  0x1D                    event-type = MDC_NOTI_SCAN_REPORT_FIXED
0x00  0x28                    event-info.length = 40
0xF0  0x00                    ScanReportInfoFixed.data-req-id =   0xF000
0x00  0x00                    ScanReportInfoFixed.scan-report-no =   0
0x00  0x02                    ScanReportInfoFixed.obs-scan-fixed.count = 2
0x00  0x20                    ScanReportInfoFixed.obs-scan-fixed.length = 32
0x00  0x01                    ScanReportInfoFixed.obs-scan-fixed.value[0].obj-handle = 1
0x00  0x0C                    ScanReportInfoFixed.obs-scan-fixed.value[0]. obs-val-data.length = 12
0xFF  0x00  0x02  0x70        Simple-Nu-Observed-Value = 62.4
0x20  0x07  0x02  0x01        Absolute-Time-Stamp = 2007-02-01T12:10:0000
0x12  0x10  0x00  0x00
0x00  0x02                    ScanReportInfoFixed.obs-scan-fixed.value[1].obj-handle = 2
0x00  0x0C                    ScanReportInfoFixed.obs-scan-fixed.value[1]. obs-val-data.length = 12
0xFF  0x00  0x01  0x00        Simple-Nu-Observed-Value = 25.6
0x20  0x07  0x02  0x01        Absolute-Time-Stamp = 2007-02-01T12:10:0000
0x12  0x10  0x00  0x00
```

H.2.4.2 Ответ на инициируемую агентом передачу данных измерений

Менеджер подтверждает получение отчета о событии от агента.

```
0xE7  0x00                    APDU CHOICE Type (PrstApdu)
0x00  0x12                    CHOICE.length = 18
0x00  0x10                    OCTET STRING.length = 16
0x43  0x21                    invoke-id =   0x4321 (start of DataApdu. MDER encoded.)
0x02  0x01                    CHOICE(Remote Operation Response | Confirmed Event Report)
0x00  0x0A                    CHOICE.length = 10
0x00  0x00                    obj-handle =   0 (MDS object)
0x00  0x00  0x00  0x00        currentTime =   0
0x0D  0x1D                    event-type = MDC_NOTI_SCAN_REPORT_FIXED
0x00  0x00                    event-reply-info.length = 0
```

H.2.4.3 Удаленное выполнение вызова подтвержденного действия по запросу данных в режиме одиночного ответа

Менеджер запрашивает измерения агента в режиме одиночного ответа.

```
0xE7  0x00                    APDU CHOICE Type (PrstApdu)
0x00  0x1E                    CHOICE.length = 30
0x00  0x1C                    OCTET STRING.length = 28
0x76  0x54                    invoke-id =   0x7654 (start of DataApdu. MDER encoded.)
0x01  0x07                    CHOICE(Remote Operation Invoke | Confirmed Action)
0x00  0x16                    CHOICE.length = 22
0x00  0x00                    obj-handle =   0 (MDS object)
0x0C  0x1B                    action-type = MDC_ACT_DATA_REQUEST
0x00  0x10                    action-info-args.length = 16
0x01  0x00                    data-req-id =   0x0100
0x84  0x80                    data-req-mode = Start | Scope Class | Mode Single Rsp
0x00  0x00  0x00  0x00        data-req-time = not used in this mode
0x00  0x00                    data-req-person-id = not used in this mode
0x00  0x06                    data-req-class = MDC_MOC_VMO_METRIC_NU
0x00  0x00  0x00  0x00        data-req-handle-list = не используется в этом режиме (count = 0, length = 0)
```

H.2.4.4 Удаленное выполнение ответа на подтвержденное действие запроса данных в режиме одиночного ответа

Агент отвечает менеджеру, высылая ему свои измерения.

```
0xE7  0x00                    APDU CHOICE Type (PrstApdu)
0x00  0x40                    CHOICE.length = 64
0x00  0x3E                    OCTET STRING.length = 62
```

0x76	0x54			invoke-id = 0x7654 (start of DataApdu. MDER encoded.)
0x02	0x07			CHOICE (Remote Operation Response | Confirmed Action)
0x00	0x38			CHOICE.length = 56
				ActionResultSimple
0x00	0x00			obj-handle = 0 (MDS object)
0x0C	0x1B			action-type = MDC_ACT_DATA_REQUEST
0x00	0x32			action-info-args.length = 50
				DataResponse
0x00	0x00	0x00	0x00 r	el-time-stamp
0x00	0x00			data-req-result = 0
0x0D	0x1D			event-type = MDC_NOTI_SCAN_REPORT_FIXED
0x00	0x28			event-info.length = 40
0x01	0x00			ScanReportInfoFixed.data-req-id = 0x0100
0x00	0x00			ScanReportInfoFixed.scan-report-no = 0
0x00	0x02			ScanReportInfoFixed.obs-scan-fixed.count = 2
0x00	0x20			ScanReportInfoFixed.obs-scan-fixed.length = 32
0x00	0x01			ScanReportInfoFixed.obs-scan-fixed.value[0].obj-handle = 1
0x00	0x0C			ScanReportInfoFixed.obs-scan-fixed.value[0]. obs-val-data.length = 12
0xFF	0x00	0x02	0x70	Simple-Nu-Observed-Value = 62.4
0x20	0x07	0x02	0x01	Absolute-Time-Stamp = 2007-02-01T12:10:0000
0x12	0x10	0x00	0x00	
0x00	0x02			ScanReportInfoFixed.obs-scan-fixed.value[1].obj-handle = 2
0x00	0x0C			ScanReportInfoFixed.obs-scan-fixed.value[1]. obs-val-data.length = 12
0xFF	0x00	0x01	0x00	Simple-Nu-Observed-Value = 25.6
0x20	0x07	0x02	0x01 A	bsolute-Time-Stamp = 2007-02-01T12:10:0000
0x12	0x10	0x00	0x00	

Н.3 Пульсовый оксиметр

Предположение:

Эта передача данных измерения инициируется менеджером, и значение поля data-req-id устанавливается менеджером равным 1.

Pleth Wave

SaSpec:: размер массива = 5,

SampleType:: объем выборки = 16,

SampleType:: значимые биты = 16,

Период дискретизации = 20 мс

→ Период обновления наблюдаемой величины RTSA = 100 мс.

Число SpO2,

Период обновления числа 1 с

→ в каждое 10-е сообщение будет включено число SpO2.

Агент отправляет девять сообщений менеджеру, которые примерно соответствуют представленному ниже формату, изменяя только информацию в сообщении (например, изменяются Invoke-id и scan-report-no).

0xE7	0x00			APDU CHOICE Type (DataApdu)
0x00	0x2A			CHOICE.length = 42
0x00	0x28			OCTET STRING.length = 40
0x43	0x21			invoke-id = 0x4321 (start of DataApdu. MDER encoded.)
0x01	0x00			CHOICE (Remote Operation Invoke | Event Report)
0x00	0x22			CHOICE.length = 34
0x00	0x00			obj-handle = 0 (MDS object)
0x00	0x00	0x00	0x00	event-time = 0
0x0D	0x1D			event-type = MDC_NOTI_SCAN_REPORT_FIXED
0x00	0x18			event-info.length = 24
0x00	0x01			ScanReportInfoFixed.data-req-id = 1
0x00	0x00			ScanReportInfoFixed.scan-report-no = 0
0x00	0x01			ScanReportInfoFixed.obs-scan-fixed.count = 1
0x00	0x0E			ScanReportInfoFixed.obs-scan-fixed.length = 14
0x00	0x01			ScanReportInfoFixed.obs-scan-fixed.value[0].obj-handle = 1 Pleth Wave
0x00	0x0C			ScanReportInfoFixed.obs-scan-fixed.value[0].obs-val-data.length = 12
0x00	0x0A			Simple-Sa-Observed-Value OCTET STRING length = 10
0xSS	0xSS	0xSS	0xSS	Samples
0xSS	0xSS	0xSS	0xSS	Samples
0xSS	0xSS			Samples

В каждое десятое сообщение агент также включает значение SpO2.

0xE7	0x00			APDU CHOICE Type (PrstApdu)
0x00	0x32			CHOICE.length = 50
0x00	0x30			OCTET STRING.length = 48
0x43	0x2A			invoke-id = 0x432A (start of DataApdu. MDER encoded.)
0x01	0x00			CHOICE (Remote Operation Invoke | Event Report)
0x00	0x2A			CHOICE.length = 42
0x00	0x00			obj-handle = 0 (MDS object)
0x00	0x00	0x00	0x00	event-time = 0
0x0D	0x1D			event-type = MDC_NOTI_SCAN_REPORT_FIXED
0x00	0x20			event-info.length = 32
0x00	0x01			ScanReportInfoFixed.data-req-id = 1
0x00	0x09			ScanReportInfoFixed.scan-report-no = 9
0x00	0x02			ScanReportInfoFixed.obs-scan-fixed.count = 2
0x00	0x16			ScanReportInfoFixed.obs-scan-fixed.length = 22
0x00	0x01			ScanReportInfoFixed.obs-scan-fixed.value[0].obj-handle = 1 Pleth Wave
0x00	0x0C			ScanReportInfoFixed.obs-scan-fixed.value[0].obs-val-data.length = 12
0x00	0x0A			Simple-Sa-Observed-Value OCTET STRING length = 10
0xSS	0xSS	0xSS	0xSS	Samples
0xSS	0xSS	0xSS	0xSS	Samples
0xSS	0xSS			Samples
0x00	0x02			ScanReportInfoFixed.obs-scan-fixed.value[1].obj-handle = 2 SpO2
0x00	0x04			ScanReportInfoFixed.obs-scan-fixed.value[1].obs-val-data.length = 4
0xFF	0x00	0x03	0xDF	Simple-Nu-Observed-Value = 99.1

Приложение I
(обязательное)

Коды номенклатуры

Данное приложение содержит коды номенклатуры, используемые в настоящем стандарте. Они скопированы из ИСО/ИИЭР 11073-10101 [12] или созданы для настоящего стандарта и включены в ИСО/ИИЭР 11073-10101.

Используемый в настоящем стандарте формат соответствует формату, определенному в ИСО/ИИЭР 11073-10101.

```
/* Коды разделов                                                                                                    */
#define MDC_PART_OBJ                          1          /* Object Infrastr.                                        */
#define MDC_PART_SCADA                        2          /* SCADA (Physio IDs                                       * /
#define MDC_PART_DIM                          4          /* Dimension                                               */
#define MDC_PART_INFRA                        8          /* Infrastructure                                          */
#define MDC_PART_PHD_DM                       128        /* Disease Mgmt                                            */
#define MDC_PART_PHD_HF                       129        /* Health and Fitness                                      */
#define MDC_PART_PHD_AI                       130        /* Aging Independently                                     */
#define MDC_PART_RET_CODE                     255        /* Return Codes                                            */
#define MDC_PART_EXT_NOM                      256        /* Ext. Nomenclature                                       */
/*****************************************************************************************************************
* В объектной инфраструктуре (MDC_PART_OBJ)
*****************************************************************************************************************/
#define MDC_MOC_VMO_METRIC                    4          /*                                                         */
#define MDC_MOC_VMO_METRIC_ENUM               5          /*                                                         */
#define MDC_MOC_VMO_METRIC_NU                 6          /*                                                         */
#define MDC_MOC_VMO_METRIC_SA_RT              9          /*                                                         */
#define MDC_MOC_SCAN                          16         /*                                                         */
#define MDC_MOC_SCAN_CFG                      17         /*                                                         */
#define MDC_MOC_SCAN_CFG_EPI                  18         /*                                                         */
#define MDC_MOC_SCAN_CFG_PERI                 19         /*                                                         */
#define MDC_MOC_VMS_MDS_SIMP                  37         /*                                                         */
#define MDC_MOC_VMO_PMSTORE                   61         /*                                                         */
#define MDC_MOC_PM_SEGMENT                    62         /*                                                         */
#define MDC_ATTR_CONFIRM_MODE                 2323       /*                                                         */
#define MDC_ATTR_CONFIRM_TIMEOUT              2324       /*                                                         */
#define MDC_ATTR_ID_HANDLE                    2337       /*                                                         */
#define MDC_ATTR_ID_INSTNO                    2338       /*                                                         */
#define MDC_ATTR_ID_LABEL_STRING              2343       /*                                                         */
#define MDC_ATTR_ID_MODEL                     2344       /*                                                         */
#define MDC_ATTR_ID_PHYSIO                    2347       /*                                                         */
#define MDC_ATTR_ID_PROD_SPECN                2349       /*                                                         */
#define MDC_ATTR_ID_TYPE                      2351       /*                                                         */
#define MDC_ATTR_METRIC_STORE_CAPAC_CNT       2369       /*                                                         */
#define MDC_ATTR_METRIC_STORE_SAMPLE_ALG      2371       /*                                                         */
#define MDC_ATTR_METRIC_STORE_USAGE_CNT       2372       /*                                                         */
#define MDC_ATTR_MSMT_STAT                    2375       /*                                                         */
#define MDC_ATTR_NU_ACCUR_MSMT                2378       /*                                                         */
#define MDC_ATTR_NU_CMPD_VAL_OBS              2379       /*                                                         */
#define MDC_ATTR_NU_VAL_OBS                   2384       /*                                                         */
#define MDC_ATTR_NUM_SEG                      2385       /*                                                         */
#define MDC_ATTR_OP_STAT                      2387       /*                                                         */
#define MDC_ATTR_POWER_STAT                   2389       /*                                                         */
#define MDC_ATTR_SA_SPECN                     2413       /*                                                         */
#define MDC_ATTR_SCALE_SPECN_I16              2415       /*                                                         */
```

```
#define MDC_ATTR_SCALE_SPECN_I32 2416 /* */
#define MDC_ATTR_SCALE_SPECN_I8 2417 /* */
#define MDC_ATTR_SCAN_REP_PD 2421 /* */
#define MDC_ATTR_SEG_USAGE_CNT 2427 /* */
#define MDC_ATTR_SYS_ID 2436 /* */
#define MDC_ATTR_SYS_TYPE 2438 /* */
#define MDC_ATTR_TIME_ABS 2439 /* */
#define MDC_ATTR_TIME_BATT_REMAIN 2440 /* */
#define MDC_ATTR_TIME_END_SEG 2442 /* */
#define MDC_ATTR_TIME_PD_SAMP 2445 /* */
#define MDC_ATTR_TIME_REL 2447 /* */
#define MDC_ATTR_TIME_STAMP_ABS 2448 /* */
#define MDC_ATTR_TIME_STAMP_REL 2449 /* */
#define MDC_ATTR_TIME_START_SEG 2450 /* */
#define MDC_ATTR_TX_WIND 2453 /* */
#define MDC_ATTR_UNIT_CODE 2454 /* */
#define MDC_ATTR_UNIT_LABEL_STRING 2457 /* */
#define MDC_ATTR_VAL_BATT_CHARGE 2460 /* */
#define MDC_ATTR_VAL_ENUM_OBS 2462 /* */
#define MDC_ATTR_TIME_REL_HI_RES 2536 /* */
#define MDC_ATTR_TIME_STAMP_REL_HI_RES 2537 /* */
#define MDC_ATTR_DEV_CONFIG_ID 2628 /* */
#define MDC_ATTR_MDS_TIME_INFO 2629 /* */
#define MDC_ATTR_METRIC_SPEC_SMALL 2630 /* */
#define MDC_ATTR_SOURCE_HANDLE_REF 2631 /* */
#define MDC_ATTR_SIMP_SA_OBS_VAL 2632 /* */
#define MDC_ATTR_ENUM_OBS_VAL_SIMP_OID 2633 /* */
#define MDC_ATTR_ENUM_OBS_VAL_SIMP_STR 2634 /* */
#define MDC_ATTR_REG_CERT_DATA_LIST 2635 /* */
#define MDC_ATTR_NU_VAL_OBS_BASIC 2636 /* */
#define MDC_ATTR_PM_STORE_CAPAB 2637 /* */
#define MDC_ATTR_PM_SEG_MAP 2638 /* */
#define MDC_ATTR_PM_SEG_PERSON_ID 2639 /* */
#define MDC_ATTR_SEG_STATS 2640 /* */
#define MDC_ATTR_SEG_FIXED_DATA 2641 /* */
#define MDC_ATTR_PM_SEG_ELEM_STAT_ATTR 2642 /* */
#define MDC_ATTR_SCAN_HANDLE_ATTR_VAL_MAP 2643 /* */
#define MDC_ATTR_SCAN_REP_PD_MIN 2644 /* */
#define MDC_ATTR_ATTRIBUTE_VAL_MAP 2645 /* */
#define MDC_ATTR_NU_VAL_OBS_SIMP 2646 /* */
#define MDC_ATTR_PM_STORE_LABEL_STRING 2647 /* */
#define MDC_ATTR_PM_SEG_LABEL_STRING 2648 /* */
#define MDC_ATTR_TIME_PD_MSMT_ACTIVE 2649 /* */
#define MDC_ATTR_SYS_TYPE_SPEC_LIST 2650 /* */
#define MDC_ATTR_METRIC_ID_PART 2655 /* */
#define MDC_ATTR_ENUM_OBS_VAL_PART 2656 /* */
#define MDC_ATTR_SUPPLEMENTAL_TYPES 2657 /* */
#define MDC_ATTR_TIME_ABS_ADJUST 2658 /* */
#define MDC_ATTR_CLEAR_TIMEOUT 2659 /* */
#define MDC_ATTR_TRANSFER_TIMEOUT 2660 /* */
#define MDC_ATTR_ENUM_OBS_VAL_SIMP_BIT_STR 2661 /* */
#define MDC_ATTR_ENUM_OBS_VAL_BASIC_BIT_STR 2662 /* */
#define MDC_ATTR_METRIC_STRUCT_SMALL 2675 /* */
#define MDC_ATTR_NU_CMPD_VAL_OBS_SIMP 2676 /* */
#define MDC_ATTR_NU_CMPD_VAL_OBS_BASIC 2677 /* */
#define MDC_ATTR_ID_PHYSIO_LIST 2678 /* */
#define MDC_ATTR_SCAN_HANDLE_LIST 2679 /* */
```

```
/* Раздел: ACT */
#define MDC_ACT_SEG_CLR                                3084        /*          */
#define MDC_ACT_SEG_GET_INFO                           3085        /*          */
#define MDC_ACT_SET_TIME                               3095        /*          */
#define MDC_ACT_DATA_REQUEST                           3099        /*          */
#define MDC_ACT_SEG_TRIG_XFER                          3100        /*          */
#define MDC_NOTI_CONFIG                                3356        /*          */
#define MDC_NOTI_SCAN_REPORT_FIXED                     3357        /*          */
#define MDC_NOTI_SCAN_REPORT_VAR                       3358        /*          */
#define MDC_NOTI_SCAN_REPORT_MP_FIXED                  3359        /*          */
#define MDC_NOTI_SCAN_REPORT_MP_VAR                    3360        /*          */
#define MDC_NOTI_SEGMENT_DATA                          3361        /*          */
#define MDC_NOTI_UNBUF_SCAN_REPORT_VAR                 3362        /*          */
#define MDC_NOTI_UNBUF_SCAN_REPORT_FIXED               3363        /*          */
#define MDC_NOTI_UNBUF_SCAN_REPORT_GROUPED             3364        /*          */
#define MDC_NOTI_UNBUF_SCAN_REPORT_MP_VAR              3365        /*          */
#define MDC_NOTI_UNBUF_SCAN_REPORT_MP_FIXED            3366        /*          */
#define MDC_NOTI_UNBUF_SCAN_REPORT_MP_GROUPED               3367   /*          */
#define MDC_NOTI_BUF_SCAN_REPORT_VAR                        3368   /*          */
#define MDC_NOTI_BUF_SCAN_REPORT_FIXED                      3369   /*          */
#define MDC_NOTI_BUF_SCAN_REPORT_GROUPED                    3370   /*          */
#define MDC_NOTI_BUF_SCAN_REPORT_MP_VAR                     3371   /*          */
#define MDC_NOTI_BUF_SCAN_REPORT_MP_FIXED                   3372   /*          */
#define MDC_NOTI_BUF_SCAN_REPORT_MP_GROUPED                 3373   /*          */
/*****************************************************************************************************
* В медицинской системе СКАДА (MDC_PART_SCADA)
*****************************************************************************************************/
#define MDC_TEMP_BODY                  19292 /*TEMPbody                        */
#define MDC_MASS_BODY_ACTUAL           57664 /*                               */
/* Partition: SCADA/Other */
#define MDC_BODY_FAT                   57676 /*                               */
/*****************************************************************************************************
* В размерностях (MDC_PART_DIM)
*****************************************************************************************************/
#define MDC_DIM_PERCENT                544 /* %                              */
#define MDC_DIM_KILO_G                 1731 /* kg                            */
#define MDC_DIM_MIN                    2208 /* min                           */
#define MDC_DIM_HR                     2240 /* h                             */
#define MDC_DIM_DAY                    2272 /* d                             */
#define MDC_DIM_DEGC                   6048 /* ºC                            */
/*****************************************************************************************************
* В коммуникационной инфраструктуре (MDC_PART_INFRA)
*****************************************************************************************************/
#define MDC_DEV_SPEC_PROFILE_PULS_OXIM                 4100   /*          */
#define MDC_DEV_SPEC_PROFILE_BP                        4103   /*          */
#define MDC_DEV_SPEC_PROFILE_TEMP                      4104   /*          */
#define MDC_DEV_SPEC_PROFILE_SCALE                     4111   /*          */
#define MDC_DEV_SPEC_PROFILE_GLUCOSE                   4113   /*          */
#define MDC_DEV_SPEC_PROFILE_HF_CARDIO                 4137   /*          */
#define MDC_DEV_SPEC_PROFILE_HF_STRENGTH               4138   /*          */
#define MDC_DEV_SPEC_PROFILE_AI_ACTIVITY_HUB           4167   /*          */
#define MDC_DEV_SPEC_PROFILE_AI_MED_MINDER             4168   /*          */
/* Помещенный за 256 позиций от начала последнего раздела: OptionalPackageIdentifiers (т.е., 8192-256). */
#define MDC_TIME_SYNC_NONE 7936 /* никакой протокол синхронизации времени не поддерживается */
#define MDC_TIME_SYNC_NTPV3            7937 /* RFC 1305 1992 Mar obs:
                                               1119,1059,958 */
#define MDC_TIME_SYNC_NTPV4            7938 /* <under development at
                                               ntp.org> */
#define MDC_TIME_SYNC_SNTPV4           7939 /* RFC 2030 1996 Oct obs:
```

```
                                                    1769 */
#define MDC_TIME_SYNC_SNTPV4330             7940 /* RFC 4330 2006 Jan obs:
                                                    2030,1769 */
#define MDC_TIME_SYNC_BTV1                  7941 /* Профиль
                                                    Bluetooth медицинского прибора*/
/*****************************************************************************************************
* В кодах возврата (MDC_PART_RET_CODE)
*****************************************************************************************************/
#define MDC_RET_CODE_UNKNOWN                1       /* Универсальный код ошибки */
/* Ошибки раздела MDC_PART_RET_CODE/объекта OBJ */
#define MDC_RET_CODE_OBJ_BUSY               1000
/* Объект занят, поэтому не может обработать запрос */
/* Ошибки раздела MDC_PART_RETURN_CODES/STORE Storage */
#define MDC_RET_CODE_STORE_EXH              2000
/* Накопитель, такой как диск, заполнен */
#define MDC_RET_CODE_STORE_OFFLN            2001
/* Накопитель, такой как диск, в автономном режиме */
```

Приложение J
(справочное)

История появления и изменения

J.1 Общие положения

Многие аспекты информационных, сервисных и коммуникационных моделей устанавливаются в других стандартах ИСО/ИИЭР 11073. Настоящее приложение перечисляет происхождение структур ASN.1, номенклатур и правил кодирования, а также любых модификаций, адаптирующих их к предметной области персональных медицинских приборов. Целевая аудитория настоящего приложения — прежде всего пользователи, которые поддерживают и гарантируют непротиворечивость стандартов ИСО/ИИЭР 11073.

J.2 Структура ASN.1

Следующие структуры ASN.1 были импортированы из ИСО/ИИЭР 11073-10201:2004 [13] без модификации:

INT-U8, INT-I8, INT-U16, INT-I16, INT-U32, INT-I32, BITS-16, BITS-32, OID-Type, PrivateOid, HANDLE, InstNumber, TYPE, AVA-Type, AttributeList, AttributeIdList, FLOAT-Type, RelativeTime, HighResRelativeTime, AbsoluteTime, OperationalState, SystemModel, ProductionSpec, ProdSpecEntry, PowerStatus, BatMeasure, NuObsValue, NuObsValueCmp, SaSpec, SampleType, SaFlags, ScaleRangeSpec8, ScaleRangeSpec16, ScaleRangeSpec32, EnumObsValue, ConfirmMode, SetTimeInvoke, SegmSelection, SegmIdList, AbsTimeRange, SegmentInfoList, SegmentInfo, ObservationScan, и TimeProtocolId.

Следующие структуры ASN.1 были импортированы из ИСО/ИИЭР 11073-10201:2004 [13] с модификацией:

NomPartition, StoSampleAlg, MeasurementStatus и EnumVal.

Остальные структуры ASN.1 были созданы специально для настоящего стандарта.

J.3 Правила кодирования медицинских приборов (MDER)

MDER импортированы из ИСО/ИИЭР 11073-10101 [12]. Большинство изменений, внесенных в импортированные правила, были редакторскими; однако, некоторые из них были преобразованы в обязательные к выполнению (например, в таблице F.1).

Оптимизация и объяснения, описанные в F.7 и F.8, выполнены специально для настоящего стандарта.

J.4 Коды номенклатуры

J.4.1 Общие положения

В J.4.2—J.4.6 описаны источники кодов, перечисленных в приложении I.

J.4.2 Коды разделов

Были добавлены четыре новых кода разделов: MDC_PART_PHD_DM, MDC_PART_PHD_HF, MDC_PART_PHD_AI и MDC_PART_RET_CODE. Остальные заимствованы из ИСО/ИИЭР 11073-10101 [12].

J.4.3 Коды объектов инфраструктуры

J.4.3.1 MDC_MOC

Все коды номенклатуры, начинающиеся с MDC_MOC_ заимствованы из ИСО/ИИЭР 11073-10101 [12].

J.4.3.2 MDC_ATTR

Были добавлены следующие коды, начинающиеся с MDC_ATTR_: MDC_ATTR_DEV_CONFIG_ID, MDC_ATTR_MDS_TIME_INFO, MDC_ATTR_METRIC_ SPEC_SMALL, MDC_ATTR_SOURCE_HANDLE_REF, MDC_ATTR_SIMP_SA_OBS_VAL, MDC_ATTR_ENUM_OBS_VAL_SIMP_OID, MDC_ATTR_ENUM_OBS_VAL_SIMP_STR, MDC_ATTR_NU_VAL_OBS_BASIC, MDC_ATTR_PM_STORE_CAPAB, MDC_ATTR_PM_ SEG_MAP, MDC_ATTR_PM_SEG_PERSON_ID, MDC_ATTR_SEG_STATS, MDC_ATTR_ SEG_FIXED_DATA, MDC_ATTR_PM_SEG_ELEM_STAT_ATTR, MDC_ATTR_SCAN_ HANDLE_ATTR_VAL_MAP, MDC_ATTR_SCAN_REP_PD_MIN, MDC_ATTR_ATTRIBUTE_VAL_MAP, MDC_ATTR_NU_VAL_OBS_SIMP, MDC_ATTR_PM_STORE_LABEL_STRING, MDC_ATTR_PM_SEG_LABEL_STRING, MDC_ATTR_TIME_PD_MSMT_ACTIVE, MDC_ ATTR_SYS_TYPE_SPEC_LIST, MDC_ATTR_METRIC_STRUCT_SMALL, MDC_ATTR_ NU_CMPD_VAL_OBS_SIMP, MDC_ATTR_NU_CMPD_VAL_OBS_BASIC, MDC_ATTR_ID_ PHYSIO_LIST. Остальные заимствованы из ИСО/ИИЭР 11073-10101 [12].

J.4.3.3 MDC_ACT

Были добавлены следующие коды, начинающиеся с MDC_ACT_: MDC_ACT_ DATA_REQUEST и MDC_ACT_SEG_TRIG_XFER. Остальные заимствованы из ИСО/ИИЭР 11073-10101 [12].

J.4.3.4 MDC_NOTI

Все коды MDC_RET_CODE были заново созданы для настоящего стандарта.

J.4.4 Медицинская система СКАДА

Был добавлен код MDC_BODY_FAT. Остальные заимствованы из ИСО/ИИЭР 11073-10101 [12].

J.4.5 Коды размерностей

Все коды размерностей заимствованы из ИСО/ИИЭР 11073-10101 [12].

J.4.6 Коды коммуникационной инфраструктуры

J.4.6.1 MDC_DEV_SPEC_PROFILE

Были добавлены следующие коды, начинающиеся с MDC_DEV_SPEC_ PROFILE_: MDC_DEV_SPEC_ PROFILE_GLUCOSE, MDC_DEV_SPEC_PROFILE_HF_ CARDIO, MDC_DEV_SPEC_PROFILE_HF_STRENGTH, MDC_DEV_SPEC_PROFILE_AI_ ACTIVITY_HUB and MDC_DEV_SPEC_PROFILE_AI_MED_MINDER. Остальные заимствованы из ИСО/ИИЭР 11073-10101 [12].

J.4.6.2 MDC_TIME_SYNC

Все коды MDC_TIME_SYNC были заново созданы для настоящего стандарта.

Приложение ДА
(справочное)

Сведения о соответствии ссылочных международных стандартов и документов национальным стандартам Российской Федерации

Таблица ДА.1

Обозначение ссылочного международного стандарта, документа	Степень соответствия	Обозначение и наименование соответствующего национального стандарта
ИИЭР 1073	—	*
ИСО/МЭК 8327-1	IDT	ГОСТ Р ИСО/МЭК 8327-1—96 «Информационная технология. Взаимосвязь открытых систем. Протокол сеансового уровня в режиме с установлением соединения. Часть 1. Спецификация протокола»
ИСО/МЭК 8650-1	—	*
ИСО/МЭК 8824-1	IDT	ГОСТ Р ИСО/МЭК 8824-1—2001 «Информационная технология. Абстрактная синтаксическая нотация версии один (АСН.1). Часть 1. Спецификация основной нотации»
ИСО/МЭК 8824-2	IDT	ГОСТ Р ИСО/МЭК 8824-2—2001 «Информационные технологии. Абстрактная синтаксическая нотация версии один (АСН.1). Часть 2: Спецификация информационного объекта»
ИСО/МЭК 8825-1	IDT	ГОСТ Р ИСО/МЭК 8825-1—2003 «Информационные технологии. Правила кодирования языка ASN.1. Часть 1. Спецификация базовых (BER), канонических (CER) и отличительных (DER) правил кодирования»
ИСО/МЭК 9072-2	IDT	ГОСТ Р ИСО/МЭК 9072-2—93 «Системы обработки информации. Передача текста. Удаленные операции. Часть 2. Спецификация протокола»
ИСО/МЭК 9595	IDT	ГОСТ Р ИСО/МЭК 9595—99 «Информационная технология. Взаимосвязь открытых систем. Определение общих сервисов информации административного управления»
ИСО/МЭК 9596-1	IDT	ГОСТ Р ИСО/МЭК 9596-1—99 «Информационная технология. Взаимосвязь открытых систем. Протокол информации административного управления. Часть 1. Спецификация протокола»
ИСО/МЭК 9899	—	*
ИСО/МЭК 11188-3	—	*
ИСО/ИИЭР 11073-10101	—	*
ИСО/ИИЭР 11073-10201	—	*
ИСО/ИИЭР 11073-30200	—	*
ИСО/ИИЭР 11073-30300	—	*

* Соответствующий национальный стандарт отсутствует. До его утверждения рекомендуется использовать перевод на русский язык данного международного стандарта. Перевод данного международного стандарта находится в Федеральном информационном фонде технических регламентов и стандартов.

Примечание — В настоящей таблице использовано следующее условное обозначение степени соответствия стандартов:

- IDT — идентичные стандарты.

Библиография

[1] Device specialization — Cardiovascular fitness and activity monitor

[2] ИИЭРР11073-10442™, Health informatics — Personal health device communication — Part 10442: Device specialization — Strength fitness equipment

[3] ИИЭРStd 11073-10408™, Health informatics — Personal health device communication — Part 10408: Device specialization — Thermometer

[4] ИИЭРStd 11073-10415™, Health informatics — Personal health device communication — Part 10415: Device specialization — Weighing scale

[5] ИИЭРStd 11073-10471™, Health informatics — Personal health device communication — Part 10471: Device specialization — Independent living activity hub

[6] ИИЭР100, The Authoritative Dictionary of ИИЭР Standards Terms, Seventh Edition, New York, Institute of Electrical and Electronic Engineers, Inc.

[7] ISO/IEC 646 (1991), Information technology — ISO 7-bit coded character set for information interchange.20

[8] ISO/ИИЭРР11073-00103 (Draft 1, 13 Aug. 2007), Health informatics — Personal health device communication — Part 00103: Technical report — Overview

[9] ISO/ИИЭРР11073-10404, Health informatics — Personal health device communication — Part 10404: Device specialization — Pulse oximeter

[10] ISO/ИИЭРР11073-10407, Health informatics — Personal health device communication — Part 10407: Device specialization — Blood pressure monitor

[11] ISO/ИИЭРР11073-10417, Health informatics — Personal health device communication — Part 10417: Device specialization — Glucose meter

[12] ISO/ИИЭР11073-10101, Health informatics — Point-of-care medical device communication — Part 10101: Nomenclature

[13] ISO/ИИЭР11073-10201:2004, Health informatics — Point-of-care medical device communication — Part 10201: Domain information model

[14] ISO/ИИЭР11073-20101:2004, Health informatics — Point-of-care medical device communication — Part 20101: Application Profiles — Base Standard

[15] ISO 15225, Nomenclature — Specification for a nomenclature system for medical devices for the purpose of regulatory data exchange

[16] ITU-T Rec. X.680 (Jul. 2002), Information technology — Abstract Syntax Notation One (ASN.1): Specification of basic notation.21

[17] ITU-T Rec. X.691 (Jul. 2002), Information technology — ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)

[18] ITU-T Rec. X.693 (Dec. 2001), Information technology — ASN.1 encoding rules: XML Encoding Rules (XER)

УДК 004:61:006.354 ОКС 35.240.80 П85 ОКСТУ 4002

Ключевые слова: здравоохранение, информатизация здоровья, структуры данных, персональные медицинские приборы, передача данных, информационное взаимодействие, прикладные профили, протокол обмена, оптимизированный протокол

Редактор *А.Ф. Колчин*
Технический редактор *В.Ю. Фотиева*
Корректор *Р.А. Ментова*
Компьютерная верстка *Е.Е. Кругова*

Сдано в набор 13.05.2016. Подписано в печать 02.06.2016. Формат 60 ×84 1/8. Гарнитура Ариал.
Усл. печ. л. 20,46. Уч.-изд. л. 19,70. Тираж 29 экз. Зак. 1399.

Издано и отпечатано во ФГУП «СТАНДАРТИНФОРМ», 123995 Москва, Гранатный пер., 4.
gost.e-ecolog.ru gost.e-ecolog.ru

gost.e-ecolog.ru