

# МЕЖДУНАРОДНЫЙ СТАНДАРТ

ISO  
5231

Первое издание  
2022-10

---

## **Межсистемный интерфейс данных информационных систем управления сельскохозяйственным производством (МИДС) – Концепция и руководство**



## ДОКУМЕНТ ЗАЩИЩЕН ПРАВАМИ ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

© ISO 2022

Все права защищены. При отсутствии иных указаний или требований в контексте применения ни одна часть данного издания не может быть воспроизведена или использована иным образом в любой форме или любым способом, в электронном или физическом виде, включая фотокопирование или размещение в Интернет или какой-либо внутренней сети без предварительного письменного разрешения. Разрешение может быть запрошено у ISO по адресу, указанному далее, или у члена ISO в стране заявителя.

Бюро регистрации авторских прав ISO

CP 401 • Ch. de Blandonnet 8

CH-1214 Вернье, Женева

Телефон: +41 22 749 01 11

Электронная почта: [copyright@iso.org](mailto:copyright@iso.org)

Веб-сайт: [www.iso.org](http://www.iso.org)

Опубликовано в Швейцарии

# Содержание

Страница

|                                                                                               |           |
|-----------------------------------------------------------------------------------------------|-----------|
| <b>Предисловие</b> .....                                                                      | <b>v</b>  |
| <b>Введение</b> .....                                                                         | <b>vi</b> |
| <b>1 Область применения</b> .....                                                             | <b>1</b>  |
| <b>2 Ссылки на нормативные документы</b> .....                                                | <b>1</b>  |
| <b>3 Термины и определения</b> .....                                                          | <b>1</b>  |
| <b>4 Обзор архитектуры</b> .....                                                              | <b>3</b>  |
| <b>5 Сеть</b> .....                                                                           | <b>4</b>  |
| 5.1 Общее описание .....                                                                      | 4         |
| 5.2 Сервис интеграции .....                                                                   | 5         |
| 5.3 Подключенные сети .....                                                                   | 5         |
| <b>6 Обмен сообщениями</b> .....                                                              | <b>6</b>  |
| 6.1 Общее описание .....                                                                      | 6         |
| 6.2 Иерархическая структура наборов сценариев, потоков выполнения сценариев и сценариев ..... | 6         |
| 6.2.1 Общая информация .....                                                                  | 6         |
| 6.2.2 Сценарий .....                                                                          | 7         |
| 6.2.3 Поток выполнения сценариев .....                                                        | 9         |
| 6.2.4 Набор сценариев .....                                                                   | 9         |
| 6.2.5 Подробное описание сценария .....                                                       | 9         |
| 6.3 Адресация .....                                                                           | 13        |
| 6.3.1 Общая информация .....                                                                  | 13        |
| 6.3.2 Возможности конечных точек .....                                                        | 13        |
| 6.3.3 Подписки конечных точек .....                                                           | 14        |
| 6.3.4 Возможности коммуникационного компонента .....                                          | 15        |
| 6.4 Архитектура конечных точек .....                                                          | 15        |
| 6.4.1 Общее описание .....                                                                    | 15        |
| 6.4.2 Ящик входящих и исходящих сообщений конечной точки .....                                | 16        |
| 6.4.3 Лента сообщений конечных точек .....                                                    | 16        |
| 6.5 Установление подключения .....                                                            | 18        |
| 6.6 Авторизация .....                                                                         | 18        |
| 6.7 Потокковая передача данных .....                                                          | 19        |
| 6.8 Разделение на фрагменты .....                                                             | 20        |
| 6.9 Обработка ошибок .....                                                                    | 20        |
| <b>7 Формальное и семантическое описание</b> .....                                            | <b>21</b> |
| 7.1 Общая информация .....                                                                    | 21        |

|          |                                                                                        |           |
|----------|----------------------------------------------------------------------------------------|-----------|
| 7.2      | Существительное.....                                                                   | 21        |
| 7.3      | Сценарий .....                                                                         | 21        |
| 7.4      | Поток выполнения сценариев .....                                                       | 22        |
| 7.5      | Набор сценариев .....                                                                  | 23        |
| 7.6      | Структура документации .....                                                           | 24        |
| 7.7      | Структура файлов буферов протокола .....                                               | 24        |
| 7.7.1    | Общая структура .....                                                                  | 24        |
| 7.7.2    | Структура кода ошибки.....                                                             | 25        |
| <b>8</b> | <b>Структурирование транспортного уровня.....</b>                                      | <b>25</b> |
| 8.1      | Общая информация.....                                                                  | 25        |
| 8.2      | Обнаружение сервиса .....                                                              | 26        |
| 8.2.1    | Общая информация .....                                                                 | 26        |
| 8.2.2    | Подключение через Интернет.....                                                        | 26        |
| 8.2.3    | Локальная сеть .....                                                                   | 27        |
|          | <b>Приложение А (справочное) Глаголы OAGIS .....</b>                                   | <b>28</b> |
|          | <b>Приложение В (справочное) Перевод элементов ISO 11783-10 в буфер протокола.....</b> | <b>29</b> |
|          | <b>Библиография.....</b>                                                               | <b>34</b> |

## Предисловие

ISO (Международная организация по стандартизации) является всемирной федерацией национальных органов по стандартизации (члены ISO). Разработка международных стандартов обычно осуществляется техническими комитетами ISO. Каждый член, заинтересованный в деятельности, для которой технический комитет был создан, имеет право быть представленным в этом комитете. Международные организации, правительственные и неправительственные, имеющие связи с ISO, также принимают участие в ее работе. ISO тесно сотрудничает с Международной Электротехнической Комиссией (IEC) по всем вопросам стандартизации в области электротехники.

Процедуры, использованные в разработке данного документа, а также предназначенные для его дальнейшей актуализации, указаны в Директивах ISO/IEC, часть 1. В частности должны быть указаны различные критерии утверждения, применимые к различным типам документов ISO. Проект данного документа был создан в соответствии с редакционными правилами Директив ISO/IEC, часть 2 (см. [www.iso.org/directives](http://www.iso.org/directives)).

Следует иметь в виду, что некоторые элементы этого документа могут быть объектом патентных прав. ISO не несет ответственность за идентификацию любого или всех таких патентных прав. Подробная информация касательно патентных прав, идентифицированных в процессе разработки данного документа, будет приведена во Введении и/или в перечне ISO, содержащем информацию о полученных патентных декларациях (см. [www.iso.org/patents](http://www.iso.org/patents)).

Любое торговое наименование, использованное в данном документе, является информацией, приведенной для удобства пользователей, и не представляет собой свидетельства в пользу какого-либо товара.

Для получения информации о рекомендательном характере стандартов, значений специальных терминов и выражений ISO, связанных с оценкой соответствия, а также информации о следовании ISO принципам Всемирной торговой организации (ВТО) в рамках Технических барьеров в торговле (ТБТ) см. [www.iso.org/iso/foreword.html](http://www.iso.org/iso/foreword.html).

Данный документ был подготовлен Техническим Комитетом ISO/TC 23, *Тракторы и машины для сельского и лесного хозяйства*, Подкомитетом SC 19, *Сельскохозяйственная электроника*.

Любая обратная связь или вопросы по данному документу следует направлять в национальный орган по стандартизации в стране пользователя. Полный перечень таких органов находится по ссылке [www.iso.org/members.html](http://www.iso.org/members.html).

## Введение

Участники сельскохозяйственной деятельности сталкиваются с необходимостью осуществлять обмен данными между различными системами и интерфейсами. С ростом спроса на контроль и управление процессами, своевременность получения информации о статусе выполнения задач и в то же время внедрение мобильных устройств в сельскохозяйственные процессы появляется потребность в стандартизации обмена данными.

# Межсистемный интерфейс данных информационных систем управления сельскохозяйственным производством (МИДС) – Концепция и руководство

## 1 Область применения

В данном документе описывается концепция расширяемой системы связи и определяются правила добавления нового функционала для обеспечения конкретных вариантов использования.

## 2 Ссылки на нормативные документы

Данный документ не содержит ссылок на нормативные документы.

## 3 Термины и определения

Для целей настоящего документа применяются следующие термины и определения.

ISO и IEC имеют терминологические базы данных по стандартизации, расположенные по следующим адресам:

- Онлайн платформа для просмотра информации ISO: по адресу <https://www.iso.org/obp>
- Электронная энциклопедия IEC: по адресу <https://www.electropedia.org/>

### 3.1 сеть

группа, состоящая из двух и более участников, связанных друг с другом через сервер

### 3.2 конечная точка

узел в сети, имеющий уникальный адрес

Примечание 1 к записи: Конечной точкой могут являться информационная система управления фермой (ИСУФ), телеметрическая установка, терминал или укомплектованная техника.

Примечание 2 к записи: Сервер также является конечной точкой.

### 3.3 клиент

С  
конечная точка, общающаяся с сервером

### 3.4 сервер

S  
центральный компонент для обеспечения связи в сети

Примечание 1 к записи: Вся связь осуществляется через сервер.

### 3.5 коммуникационный компонент

МС  
элемент управления сервера и сети

Примечание 1 к записи: Коммуникационный компонент (МС) отслеживает авторизованные конечные точки и их возможности, а также ответственен за доставку и переадресацию сообщений.

**3.6**  
**существительное**  
тип сообщения

**3.7**  
**глагол**  
действие, выполняемое по отношению к конкретному существительному

**3.8**  
**сценарий**  
порядок выполнения запроса и ответных сообщений

Примечание 1 к записи: Каждые запрос и ответ содержат глагол и существительное.

**3.9**  
**поток выполнения сценариев**  
последовательность выполнения сценариев

Примечание 1 к записи: Потоки выполнения сценариев определяют взаимосвязь между сценариями и то, каким образом они должны выполняться относительно друг друга.

**3.10**  
**набор сценариев**  
группа потоков выполнения сценариев

**3.11**  
**Шаг А**  
запрос, отправляемый от клиента А клиенту Б

**3.12**  
**Шаг В**  
запрос, отправляемый от клиента Б клиенту А

**3.13**  
**потокковая передача**  
подписка на сообщения определенного типа и последующее получение информации без запроса

**3.14**  
**интеграция**  
первоначальный доступ или регистрация конечной точки

**3.15**  
**сервис интеграции**  
сервис, обеспечивающий доступ клиентов к сети

**3.16**  
**буферы протокола**  
**protobuf**  
структуры данных, которые могут быть сериализованы в краткой двоичной форме

Примечание 1 к записи: В данном документе указание версии 3 буферов протокола (proto3) применимо ко всем определениям. См. также <https://developers.google.com/protocol-buffers/docs/proto3>

**3.17**  
**протокол организации очереди сообщений**  
**MQTT**  
стандартизированный протокол передачи данных, работающий по принципу издатель-подписчик

Примечание 1 к записи: В данном документе указание MQTT 3.1.1 применимо ко всем определениям. См. также <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html>

**3.18****протокол защищенной передачи гипертекста  
HTTPS**

набор правил защищенной передачи данных по сети Интернет или локальной сети

Примечание 1 к записи: См. также <https://datatracker.ietf.org/doc/html/rfc2818>

**3.19****семантическое версионирование**

концепция назначения номеров версий программного обеспечения

Примечание 1 к записи: Семантическое версионирование позволяет указать информацию о совместимости в номере версии, см. также <https://semver.org/>.

**3.20****база данных VDMA**

содержит все определения ISO 5231 для наборов сценариев, потоков выполнения сценариев, сценариев и сообщений.

Примечание 1 к записи: База данных находится по адресу: <https://isobus.net/isobus/efdi>

**3.21****защита транспортного уровня****TLS**

протокол безопасной связи по Интернет

**3.22****сертификаты защиты транспортного уровня****TLS-CERTS**

сертификаты, предоставляющие информацию для кодирования связи

**3.23****расширяемый протокол обмена сообщениями и информацией о присутствии****XMPP**

прикладной профиль для обмена данными

**3.24****полностью определенное имя домена****FQDN**

место с указанием всех доменных уровней

**4 Обзор архитектуры**

Межсистемный интерфейс данных управления сельскохозяйственным производством состоит из набора уровней, надстроенных на прикладном уровне сети, как показано на [Рисунке 1](#).

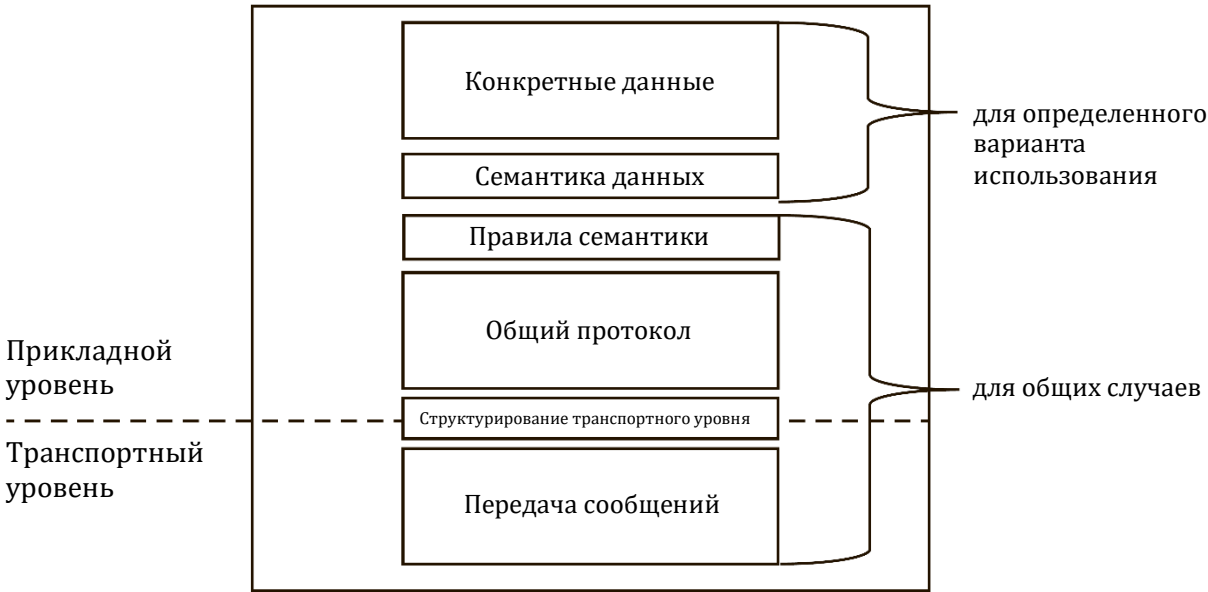


Рисунок 1 – Уровневая модель

Выбранный транспортный протокол передачи сообщений для иллюстрации реализации МИДС основан на протоколе MQTT, применяемом в качестве транспортного уровня для сообщений. Тем не менее прикладные уровни (см. [Рисунок 1](#)) не привязаны к нему. Сообщения включают заголовок и полезные данные. В заголовке содержится адресная информация, и могут находиться дополнительные данные для настройки устойчивой сессии с сохранением данных. В зависимости от функционала транспортного протокола на уровнях ниже дополнительные данные могут отсутствовать, и тогда они сообщаются на уровне транспортного протокола. Теоретически возможно передать любые полезные данные в рамках протокола МИДС. Несмотря на это передача данных согласно стандарту ISO 11783-10 подробно описана в [Приложении В](#). Также имеется общая процедура передачи файлов, описываемая набором сценариев базовой файловой функциональности. Помимо уровня для определенного варианта использования, находящегося на самом верху иерархии, МИДС имеет семантический уровень для указания получателям сообщений, что им требуется сделать с данными (выполнить задачу, заменить элемент внутри задачи и пр.).

Описание уровня общего протокола приводится в [пункте 6](#). Правила семантики описаны в [пункте 7](#), а структурирование транспортного уровня – в [пункте 8](#).

5 Сеть

5.1 Общее описание

В данном пункте представлено описание различных компонентов, составляющих элементы сети согласно [Рисунку 2](#).

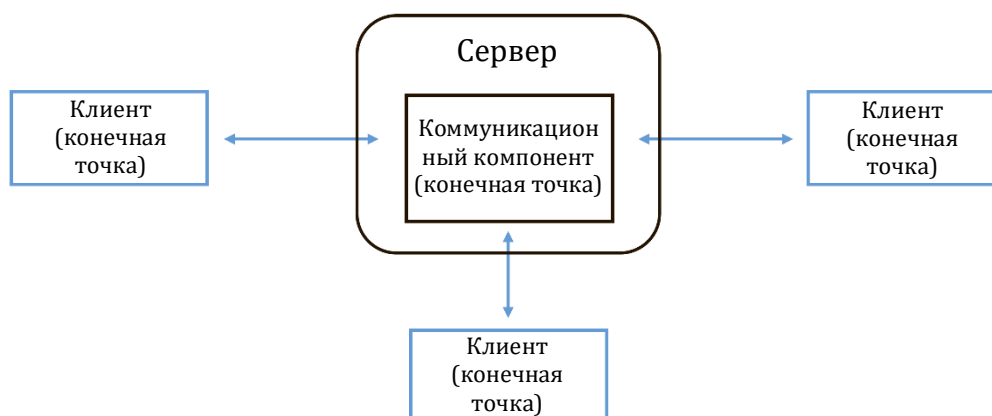


Рисунок 2 – Компоненты сети

Клиенты могут устанавливать связь с сервером. Все конечные точки сети могут связываться друг с другом.

## 5.2 Сервис интеграции

Сервис интеграции (СИ) является одним из серверных компонентов сети. Это сервис, с помощью которого конечная точка подключается к сети. При этом генерируются уникальные идентификаторы и сертификаты, необходимые для последующей связи. Интеграция является частью процесса регистрации конечной точки.

## 5.3 Подключенные сети

Подключение нескольких сетей может быть обеспечено путем внедрения сервера, который имеет как коммуникационный компонент, так и клиент.

На [Рисунке 3](#) показано, как сервер А взаимодействует с прямо подключенными клиентами через свой коммуникационный компонент. Кроме того, сервер А подключается к серверу В в качестве клиента. Через это подключение сервер А опосредованно соединяется с клиентами через сервер В. В зависимости от назначенных возможностей клиента сервера А сервер В также может иметь способность опосредованно соединяться с клиентами сервера А.

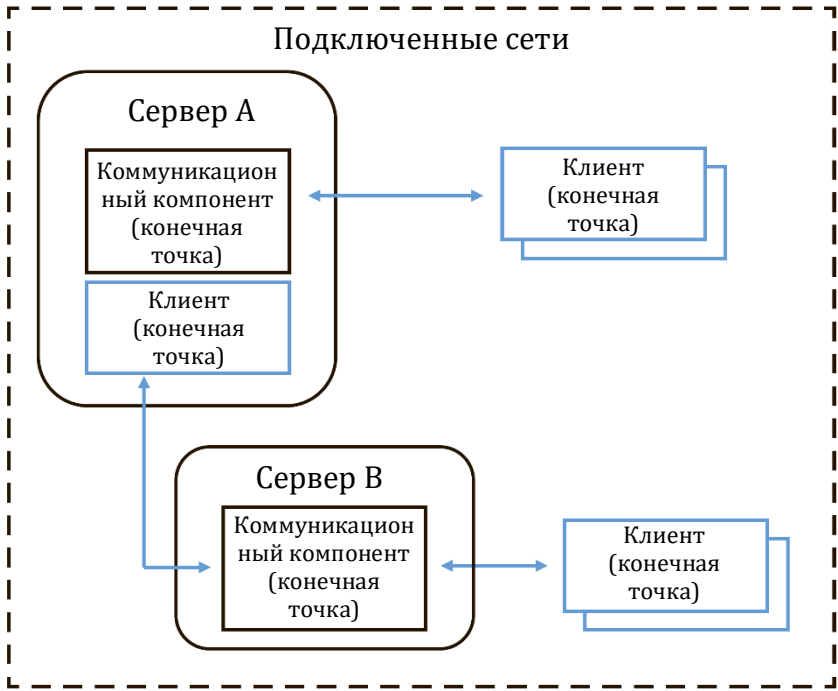


Рисунок 3 – Компоненты подключенных сетей

6 Обмен сообщениями

6.1 Общее описание

В данном пункте описаны различные основные концепции осуществления связи. Пункт включает описание структуры сообщений, архитектурных структур, адресации и потоковой передачи.

6.2 Иерархическая структура наборов сценариев, потоков выполнения сценариев и сценариев

6.2.1 Общая информация

На [Рисунке 4](#) представлена иерархическая структура наборов сценариев, потоков выполнения сценариев и сценариев. В целом, набор сценариев является высшим элементом и включает как минимум один поток выполнения сценариев. Каждый поток выполнения сценариев включает как минимум один сценарий. Сценарий состоит из запроса и ответа. И запрос, и ответ содержат комбинацию глагола и существительного.

Документы предоставляются с целью разработки национальных стандартов. Для ТК 284.

Документы предоставляются с целью разработки национальных стандартов. Для ТК 284.

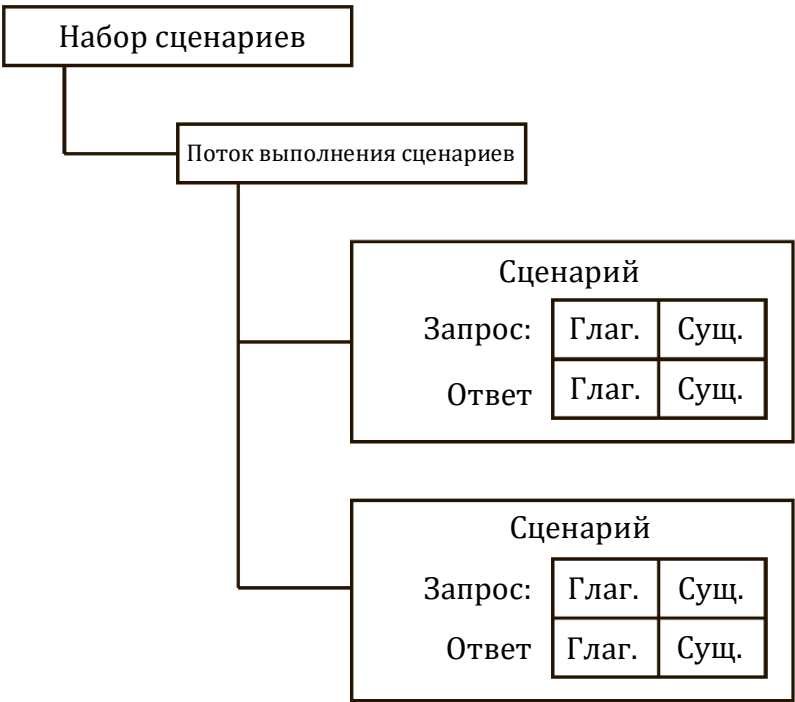


**Рисунок 4 Иерархическая структура наборов сценариев, потоков выполнения сценариев и сценариев**

Весь описанный обмен сообщениями следует общему процессу. Этот процесс описан в пунктах [6.2.2](#) – [6.2.5](#).

**6.2.2 Сценарий**

Последовательность сообщений с запросом и ответом называется сценарием. Все сценарии имеют шаблон сообщений с запросом и ответом. Это означает, что на конкретный запрос есть определенный ответ. Запрос отправляется конечной точкой, а ответ отправляется обратно другой конечной точкой.

Запросы и ответы определяются как комбинация глагола и существительного. Такая концепция «глагол-существительное» основана на Документе бизнес-объекта, определенном Спецификацией Open Applications Group по интеграции (OAGIS). См. подробно в [Приложении А](#). Глагол означает действие или действия, желательные в отношении включенной бизнес-информации. Сама бизнес-информация содержится в другом подразделе, называемом существительным.

Примером может служить сценарий запроса перечня конечных точек, подключенных к серверу, как показано на [Рисунке 5](#). Запрос содержит глагол GET (получить) и существительное ListEndpoints (перечень конечных точек). В переводе это означает запрос перечня конечных точек. Ответ содержит глагол SHOW (показать) и существительное ListEndpointsResponse (ответ, перечень конечных точек). В переводе это означает отправку перечня конечных точек.



Рисунок 5 – Сценарий ListEndpoints

Схематическое изображение показано на [Рисунке 6](#). Дополнительная информация и правила формального описания сценариев представлены в пункте [8.2](#).

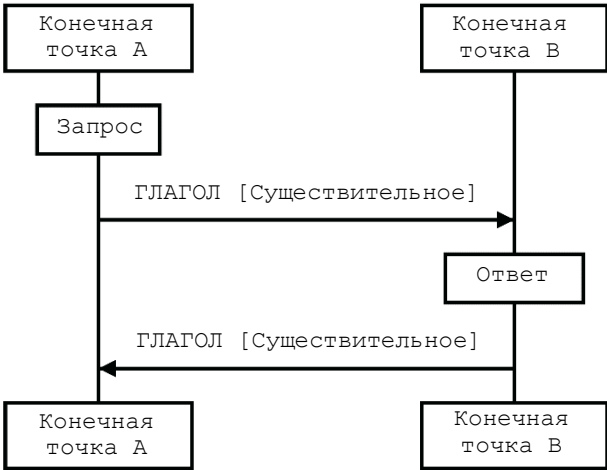


Рисунок 6 – Схематическое изображение сценария

Перечень доступных глаголов согласно стандарту OAGIS был расширен для включения новых вариантов использования. Были добавлены следующие глаголы:

- FORWARD (переадресовать): глагол должен использоваться только сервером, сообщает, что сообщение было получено и будет переадресовано другой конечной точке. Более подробное описание глагола и случаев его применения дано в пункте [6.2.5](#);
- STREAM\_START (начать потоковую передачу): для запроса потоковой передачи данных, сообщает, что конечная точка с настоящего момента желает получать потоковые данные;
- STREAM\_STOP (прекратить потоковую передачу): сообщает, что конечная точка больше не желает получать потоковые данные;
- STREAM\_DATA (передавать потоковые данные): глагол необходим для маркировки потоковых данных.

Более подробная информация о новых глаголах, применимых к потоковым данным, представлена в пункте [7.6](#).

### 6.2.3 Поток выполнения сценариев

Поток выполнения сценариев включает как минимум один сценарий. Порядок сценариев определен внутри потока. Сценарии внутри потока выполнения сценариев могут быть опциональными. Если сценарии признаются опциональными, они не должны выполняться.

Отправитель первого запроса в потоке выполнения сценариев именуется конечной точкой А, а другая конечная точка именуется конечной точкой В в рамках всей адресной коммуникации в потоке выполнения сценариев. Процесс, когда А отправляет запрос В, называется шагом А. Процесс, когда В отправляет запрос А, называется шагом В. Это показано на [Рисунке 7](#). Дополнительная информация и правила формального описания потоков выполнения сценариев представлены в пункте [7.4](#).

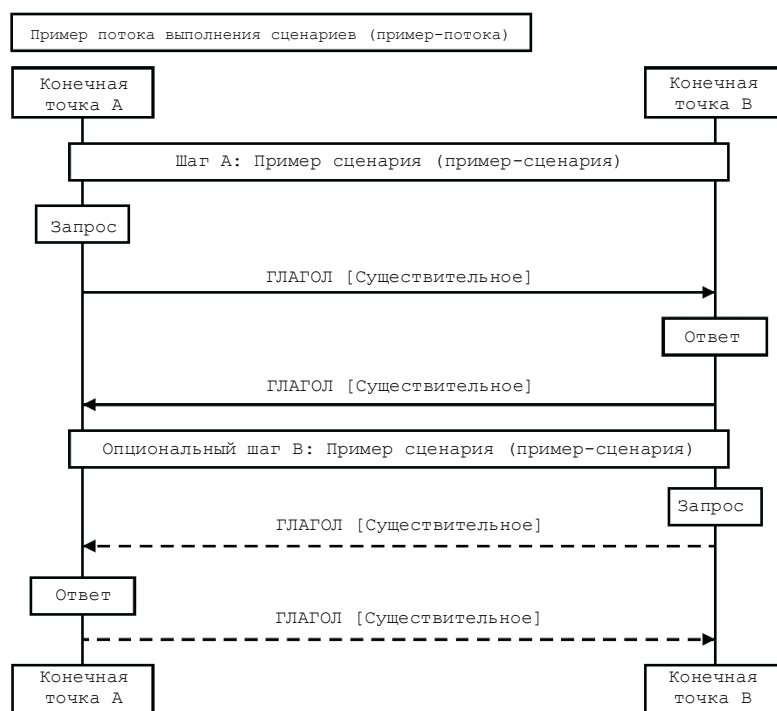


Рисунок 7 – Схематическое изображение потока выполнения сценария

### 6.2.4 Набор сценариев

Набор сценариев включает как минимум один поток выполнения сценариев. Набор сценариев — это группа, которая может применяться, например, для сертификации в случае необходимости. Дополнительная информация и правила формального описания потоков выполнения сценариев представлены в пункте [7.5](#).

В настоящий момент задан только один набор сценариев. Такой набор сценариев называется основным, и он содержит все потоки выполнения сценариев, необходимые для осуществления связи.

### 6.2.5 Подробное описание сценария

Пример сценария с прикладной точки зрения дан на [Рисунке 7](#). Запрос отправляется одной конечной точкой, а ответ отправляется обратно другой конечной точкой. Если же взглянуть на процесс более детально, мы увидим все иначе. Конечные точки не общаются непосредственно друг с другом так, как это описано в пункте [5.1](#). Вместо этого конечные точки обмениваются сообщениями друг с другом, используя сервер/ коммуникационный компонент. Упрощенная модель, состоящая из одной пары «запрос-ответ», на самом деле включает четыре подсценария. На

[Рисунке 8](#) показаны подсценарии, которые неявно выполняются в сценариях, в которых происходит обмен информацией между двумя конечными точками через сервер/коммуникационный компонент.

Эти четыре подсценария также показаны на [Рисунке 8](#):

- Подсценарий запроса конечной точки (Подсценарий\_1)
  - Запрос (от конечной точки А к серверу): ГЛАГОЛ [Существительное]
  - Ответ (от сервера к конечной точке А): FORWARD [ForwardInfo]
- Подсценарий переадресации запроса конечной точки (Подсценарий\_2)
  - Запрос (от сервера к конечной точке В): ГЛАГОЛ [Существительное]
  - Ответ (от конечной точки В к серверу): CONFIRM [ ]
- Подсценарий ответа конечной точки (Подсценарий\_3)
  - Запрос (от конечной точки В к серверу): ГЛАГОЛ [Существительное]
  - Ответ (от сервера к конечной точке В): FORWARD [ForwardInfo]
- Подсценарий переадресации конечной точки (Подсценарий\_4)
  - Запрос (от сервера к конечной точке А): ГЛАГОЛ [Существительное]
  - Ответ (от конечной точки А к серверу): CONFIRM [ ]

Элементы [Рисунка 8](#) имеют подробное описание.

- 1) Конечная точка А отправляет запрос сценария согласно определению сценария. Этот запрос отправляется на сервер. Сервер напрямую отправляет сообщение ForwardInfo (переадресация информации) в качестве ответа. Ответ сервера помимо всего прочего содержит информацию о том, что сообщение было переадресовано.
- 2) Сервер отправляет сообщение конечной точке В. Данное сообщение содержит глагол и существительное из сообщения, полученного сервером от конечной точки А. Получение этого сообщения должно подтверждаться получателем – конечной точкой В. Подтверждение содержит глагол CONFIRM (подтвердить) без существительного. Сообщение CONFIRM предназначено для подтверждения получения сообщения, которое было передано между сервером и конечной точкой А. В итоге, конечная точка А получит в качестве ответа конечной точки В сообщение, содержащее глагол и существительное.
- 3) Конечная точка В создает ответное сообщение согласно определению сценария. Такое сообщение будет отправлено на сервер в качестве сообщения запроса. Сервер напрямую отправляет сообщение ForwardInfo (переадресация информации) в качестве ответа. Ответ сервера помимо всего прочего содержит информацию о том, что сообщение было переадресовано.
- 4) Сервер отправляет сообщение конечной точке А. Данное сообщение содержит глагол и существительное из сообщения, полученного сервером от конечной точки В. Получение этого сообщения должно подтверждаться получателем – конечной точкой А. Подтверждение содержит глагол CONFIRM без существительного. Сообщение CONFIRM предназначено для подтверждения получения сообщения, которое было передано между сервером и конечной точкой В.

Все эти шаги могут быть исключены из сценария, так как они относятся к связи на уровне ниже определения сценария. По этой причине сценарии всегда представлены в упрощенной форме, как это показано на [Рисунке 7](#). Тем не менее все упомянутые подсценарии являются необходимыми для успешного обеспечения связи.

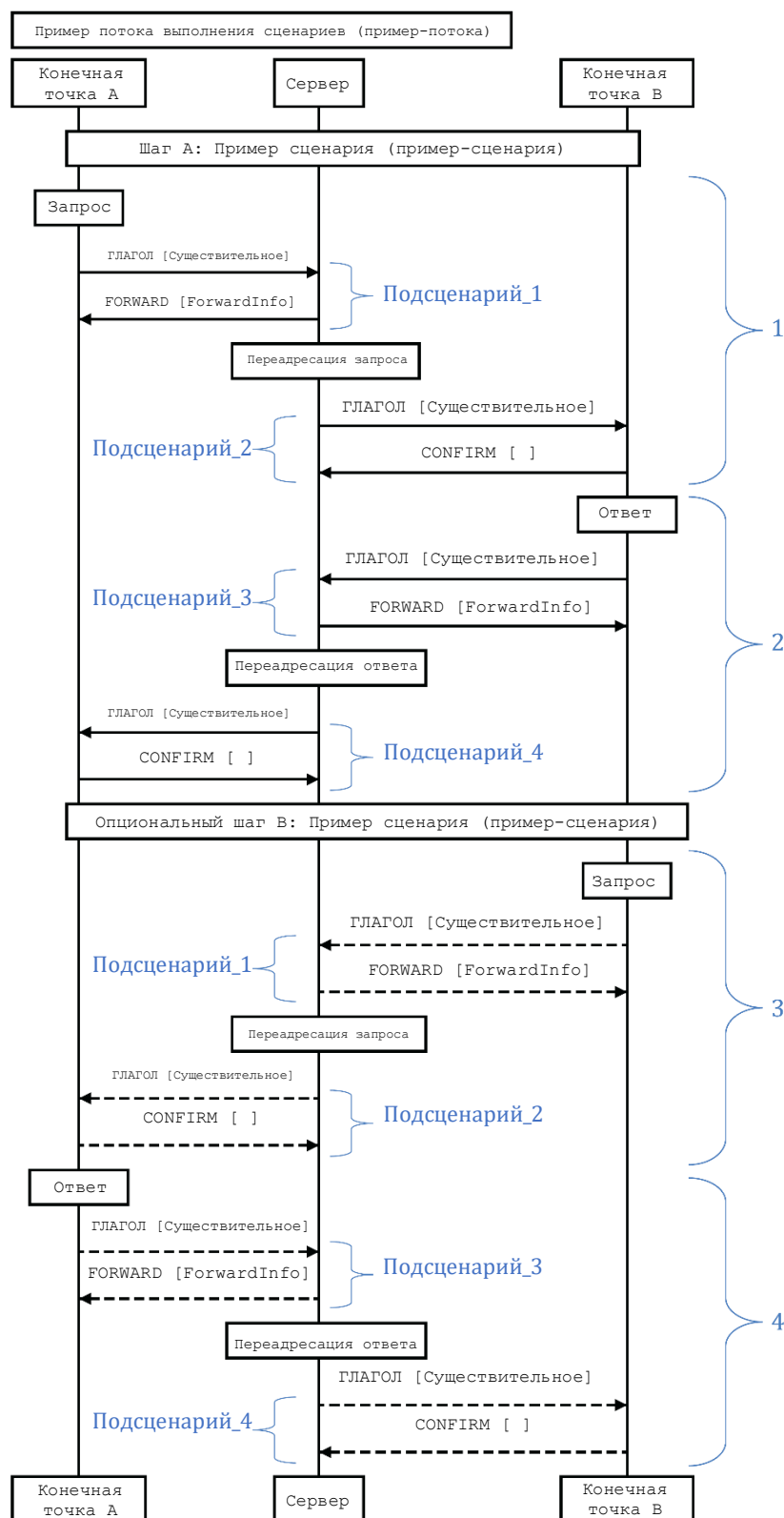


Рисунок 8 – Подробная схема потока выполнения сценариев

В сценарии, в котором ответ содержит глагол CONFIRM, ответ не будет передан в конечную точку, которая отправила запрос согласно этому сценарию. В таком случае сообщение с глаголом FORWARD должно рассматриваться как подтверждение успешной передачи. В противном случае

сообщение CONFIRM будет передано, это будет подтверждено сообщением CONFIRM и так далее. На [Рисунке 9](#) представлены подсценарии, в которых в ответе используется глагол CONFIRM.

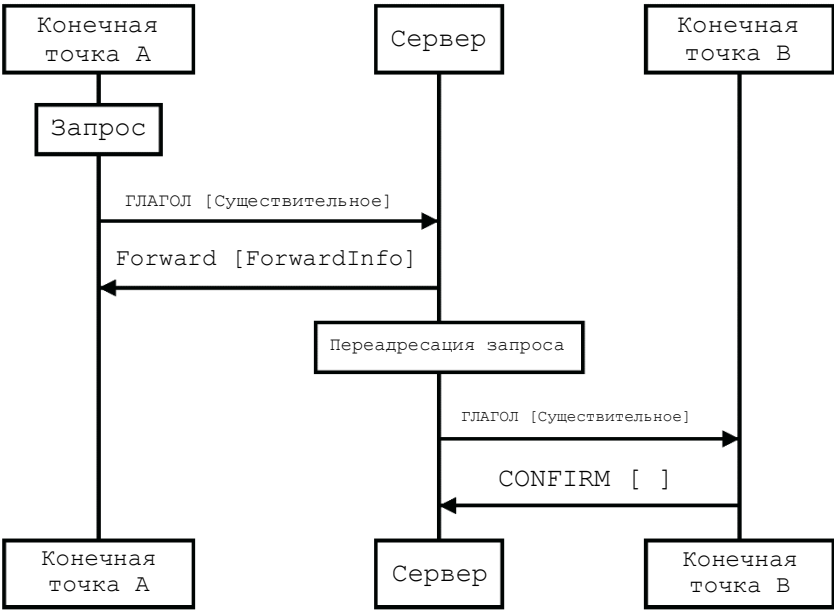


Рисунок 9 – Подробная схема сценария с ответом CONFIRM

С точки зрения сообщений, передаваемых между конечными точками и сервером, подробная схема вновь отличается. В таком случае сообщения отправляются так, как это было определено сценарием. Конечная точка должна подтверждать глаголом CONFIRM только получение последнего сообщения, отправленного сервером. Это представлено на [Рисунке 10](#). Этот процесс был несколько изменен по сравнению со сценариями, описанными для процессов между конечными точками. См. [Рисунок 9](#). Если коммуникация происходит непосредственно от конечной точки к серверу, серверу более не нужно передавать сообщение на любую иную конечную точку. По этой причине сервер не отправляет ответы, используя глагол FORWARD. Вместо этого ответ сервера на сценарный запрос должен рассматриваться в качестве подтверждения получения.

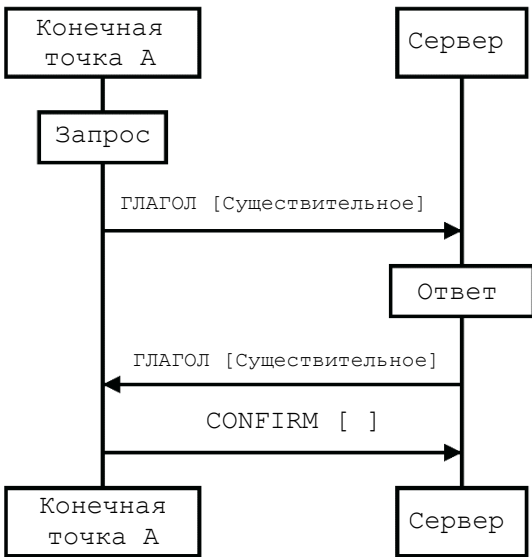


Рисунок 10 – Подробная схема сценария с участием сервера

## 6.3 Адресация

### 6.3.1 Общая информация

В данном подпункте описывается концепция уровня «Общий протокол» [Рисунка 1](#). Определения конкретных сообщений приведены в [Приложении В](#).

В целом, каждая конечная точка получает уникальный адрес. С таким адресом каждая конечная точка может быть однозначно идентифицирована. Адресация управляется коммуникационным компонентом. Основная концепция заключается в отправке наименьшего количества сообщений без предварительного запроса так, чтобы никакая информация не была изначально отправлена конечной точкой.

Сообщения могут быть адресованы напрямую одной или нескольким конечным точкам. Также они могут публиковаться, чтобы сообщение было отправлено всем подписанным конечным точкам. Сообщение может быть помечено как сообщение, которое отправляется напрямую, публикуется, или могут использоваться обе техники. Сообщения не доставляются одной и той же конечной точке дважды.

Отправка сообщений может выполняться в нескольких режимах:

- DIRECT (прямая передача): для направленной передачи в конкретную конечную точку, указать требуемых получателей
- PUBLISH (публикация): для ненаправленной передачи сообщение доставляется всем подписанным конечным точкам, получатели не указываются
- PUBLISH\_WITH\_DIRECT (публикация и прямая передача): комбинация режимом DIRECT и PUBLISH, сообщения будут доставлены всем подписанным конечным точкам, а также всем требуемым получателям, указанным в сообщении

### 6.3.2 Возможности конечных точек

У каждой конечной точки имеется ряд характеристик и способностей, которые далее именуется возможностями. Чтобы коммуникационный компонент знал, какие сообщения могут быть отправлены и получены от конечных точек, каждая конечная точка должна предоставить информацию о своих возможностях. Возможности также основаны на концепции «существительное-глагол», которая была описана в пункте [6.2.2](#). В общих словах, возможности конечной точки определяют, какие сценарии поддерживает данная конечная точка. Сценарии, не поддерживаемые конечной точкой, не могут быть получены этой конечной точкой, а конечная точка не имеет разрешения на отправку таких сообщений. Коммуникационный компонент не должен отправлять на конечную точку сообщения, которые такая конечная точка не поддерживает. Кроме того, коммуникационный компонент не должен передавать сообщения, которые конечная точка не может отправлять исходя из ее возможностей.

Когда клиенту необходимо объявить о поддержке сценария, клиент должен поддерживать запросы и ответы, определяемые конкретным сценарием. Каждые запрос и ответ содержат комбинацию глагола и существительного. Пример переведен на [Рисунке 11](#). Существуют два типа конечных точек: конечная точка А и конечная точка В. Чтобы быть отправителем запроса и получателем ответа (конечная точка А), клиент должен иметь возможность SEND – отправлять комбинацию глагола и существительного запроса и RECEIVE – получать комбинацию существительного и глагола ответа. Чтобы быть получателем запроса и отправителем ответа (конечная точка В), клиент должен иметь возможность RECEIVE – получать комбинацию существительного и глагола запроса и SEND – отправлять комбинацию глагола и существительного ответа. На основании такой информации клиент может определить, какие сценарии поддерживаются какими-либо клиентами.

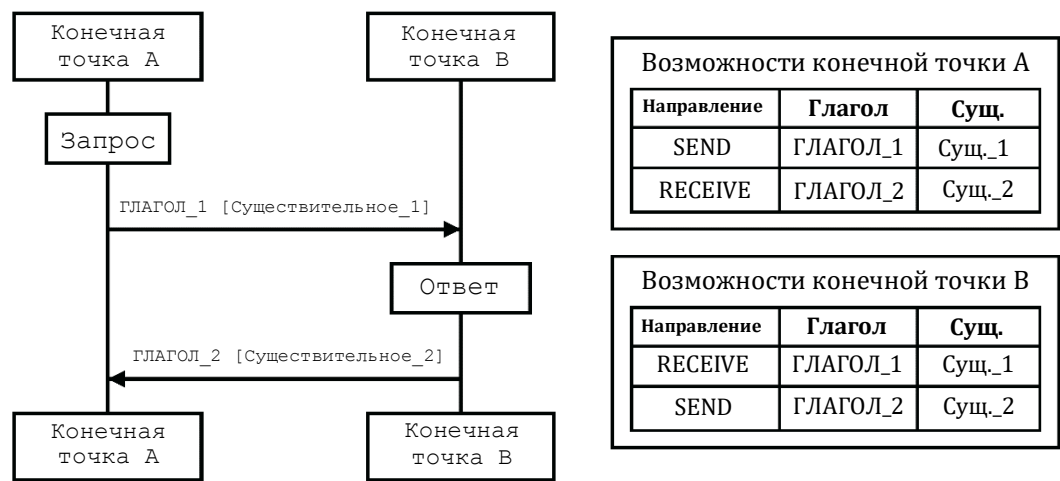


Рисунок 11 – Необходимые возможности для поддержки сценария

Информация о возможностях не отправляется без предварительного запроса. Вместо этого отправляется сообщение, содержащее данные о статусе конечной точки. Это сообщение отправляется, когда конечная точка подключается к сети, и содержит ссылку на возможности конечной точки. Если коммуникационному компоненту уже известны эта уникальная ссылка и возможности, на которые она указывает, компоненту уже известны характеристики конечной точки. Если коммуникационному компоненту еще не известны эта уникальная ссылка и связанные возможности, он может запросить их. Когда конечная точка подключается к сети, она должна отправить сообщение о статусе, обозначающее, что точка в сети. Конечная точка всегда должна создавать одинаковую ссылку на определенный набор возможностей. И наоборот, это также означает, что итоговая ссылка должна быть отличной для отличных возможностей конечной точки.

В некоторых обстоятельствах сообщение с возможностями может быть достаточно объемным. Все сценарии проходят семантическое версионирование. Если конечная точка в данный момент поддерживает несколько версий и сценариев, сообщение с возможностями становится очень большим. Если такое сообщение отправляется всегда, когда конечная точка подключается к сети, без запроса на это, создается ненужный трафик.

Если конечная точка подключилась к сети, она должна отправить отчет о том, что она в сети. Если она регулярно отключается от сети, она должна отправить отчет о том, что она по плану не будет подключена к сети. В случае, если конечная точка произвольно теряет связь с сетью, она должна сообщить о том, что она будет неожиданно не в сети.

### 6.3.3 Подписки конечных точек

Подписки используются для подписки на перечень глагола и существительных (сценариев). Публикуемые сообщения (режимы PUBLISH или PUBLISH\_WITH\_DIRECT) передаются на конечные точки, подписанные на комбинацию глагола и существительного, содержащуюся в сообщении. Подписки могут быть заданы по определенным существительным и глаголам. Каждый новый перечень подписок, отправляемый конечной точкой, удаляет старые подписки. После изменения возможностей конечная точка должна повторно отправить свои подписки. Любое сообщение об отписке удалит подписки из перечня.

Так как отправитель сообщения не всегда знает, на какие конечные точки оно отправляется, он может отправить сообщение коллективно. Любая другая конечная точка может подписаться на любой тип сообщений, который соответствует ее возможностям. Конечная точка не может подписаться на сценарии, которые она не поддерживает из-за своих возможностей.

### 6.3.4 Возможности коммуникационного компонента

Как и конечные точки коммуникационный компонент обладает возможностями. Тем не менее они несколько отличаются от возможностей конечных точек. Возможности коммуникационного компонента определяют не столько поддерживаемые сценарии, а преимущественно то, как долго и в каком объеме коммуникационный компонент способен хранить сообщения. Это так называемые минимальные возможности коммуникационного компонента, определенные в MinimumCapabilities сообщения.

Возможности конечной точки содержат информацию о том, сколько и как долго могут храниться сообщения. Также эта информация указывает на объем памяти коммуникационного компонента. Коммуникационный компонент не обязательно должен сообщать всю эту информацию, однако чем больше информации известно о коммуникационном компоненте, тем более определенным будет его поведение и тем лучше к этому смогут адаптироваться конечные точки. По крайней мере следующая информация должна быть сообщена:

- количество сообщений;
- период хранения;
- объем хранилища;
- опционально могут быть указаны максимальный размер сообщения и максимальное количество отдельных сообщений в одном общем сообщении. Возможности коммуникационного компонента по умолчанию являются минимальными – MinimumCapabilities;
- ListEndpoints (опционально).

StreamingCapabilities (опционально). В дополнение к минимальным возможностям коммуникационного компонента имеются опциональные возможности, которые может поддерживать коммуникационный компонент. Механизм перечисления конечных точек и потоковой передачи. В случае поддержки потоковой передачи все глаголы, упомянутые в пункте 6.7 могут использоваться для обмена данными с коммуникационным компонентом. Если потоковая передача не входит в возможности, она невозможна для данного коммуникационного компонента.

Каждый клиент может запросить данные о возможностях коммуникационного компонента. Адресация коммуникационному компоненту всегда происходит в режиме DIRECT, при этом перечень получателей должен быть пуст. Процедура запроса данных о возможностях описана в потоке выполнения сценариев с возможностями коммуникационного компонента. Сообщение ListEndpoints содержит запись с пустым адресом конечной точки для каждого сценария, поддерживаемого коммуникационным компонентом.

В присоединенных сетях (как показано на [Рисунке 3](#)) сервер А может подключиться к серверу В. Клиент сервера А, подключенного к серверу В, должен объявить о том, что он является частью сети, не подсоединенной к серверу В напрямую. Для этого клиент должен указать в своих возможностях, что он поддерживает сценарий Send List Endpoints. Он должен иметь возможность RECEIVE (принять) глагол GET (получить) с существительным ListEndpoints и SEND (отправить) глагол SHOW (показать) с существительным ListEndpointsResponse. Кроме того, этот клиент должен иметь возможность поддержки других сценариев, упомянутых в потоке выполнения сценариев с возможностями коммуникационного компонента.

## 6.4 Архитектура конечных точек

### 6.4.1 Общее описание

Архитектура конечных точек направлена на получение и переадресацию сообщений с учетом комбинации возможностей, подписок, публичной или прямой передачи сообщений. То, как эта

архитектура может быть структурирована в виде транспортного протокола, описано в [пункте 8](#).

6.4.2 Ящик входящих и исходящих сообщений конечной точки

На сервере имеются ящики входящих и исходящих сообщений для каждой конечной точки. Конечная точка связывается с ящиком входящих, когда сообщение отправляется на сервер. Для получения сообщения с сервера конечная точка связывается со своим ящиком исходящих. Например, если запрос сценария отправляется от конечной точки, сообщение попадает в ящик входящих. Ответ этого сценария получается через ящик исходящих. Схема того, как эта концепция реализована внутри сервера и как сообщения перемещаются между конечными точками, показана на [Рисунке 12](#).

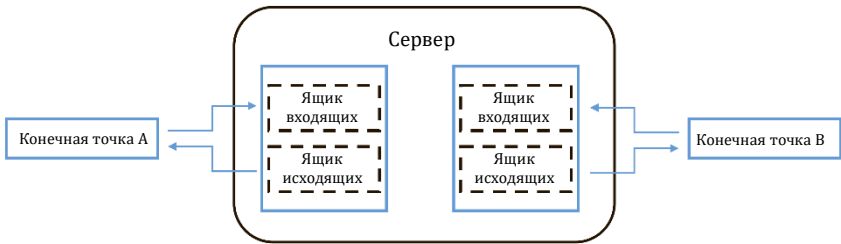


Рисунок 12 – Архитектура конечных точек в сети

На [Рисунке 13](#) показано, как сообщение отправляется от конечной точки А к конечной точке В. Сообщение отправляется от конечной точки А в ящик входящих. Внутри сервера оно переадресуется конечной точке В, попадая в ящик входящих конечной точки В. Это сообщение может быть извлечено оттуда конечной точкой В.

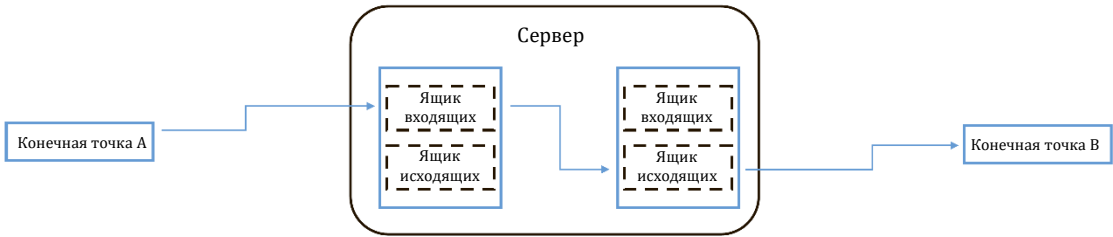


Рисунок 13 – Отправка сообщения от конечной точки А к конечной точке В

Это минимальная возможная архитектура. Внутри серверного компонента должны иметься ящики входящих и исходящих сообщений для каждой конечной точки.

6.4.3 Лента сообщений конечных точек

6.4.3.1 Общее описание

Дополнительно возможно наличие ленты сообщений для каждой конечной точки внутри серверного компонента. Эта лента используется как хранилище сообщений. Наличие или отсутствие ленты для каждой конечной точки можно проверить по возможностям коммуникационного компонента. Возможности коммуникационного компонента также могут использоваться для определения объема памяти и того, как много и как долго сообщения могут храниться. По достижении максимального объема хранилища сервера (если превышены количество хранимых сообщений или их размер для определенной конечной точки) самые старые сообщения должны быть сброшены и удалены из ленты конечной точки.

На [Рисунке 14](#) изображено, как можно схематически показать ящик входящих, ящик исходящих и ленту сообщений конечной точки на сервере. Коммуникация по-прежнему осуществляется через

ящик входящих/исходящих. Единственное отличие заключается в том, что данные ящика исходящих могут храниться в ленте.

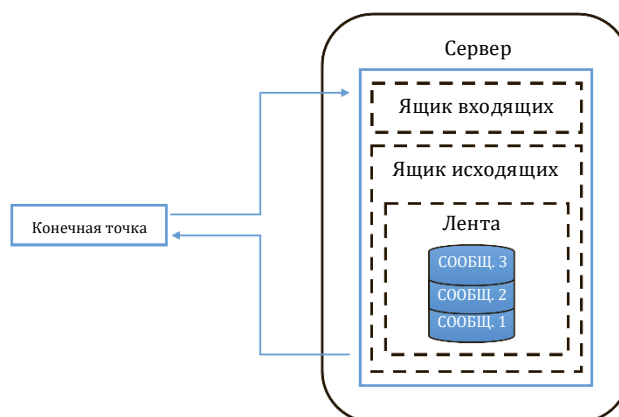


Рисунок 14 – Архитектура конечных точек с лентой

В [пункте 8](#) подробно описано, как выглядит коммуникация по запросу сообщений из ленты на основе MQTT.

#### 6.4.3.2 Конфигурация ленты

После начального установления соединения (так называемой интеграции) никакие сообщения других конечных точек не будут отправляться на интегрированную конечную точку. Эта конечная точка должна отправить данные о своих возможностях и настроить конфигурацию ленты. После этого могут быть получены сообщения от других конечных точек. Конечная точка появляется в перечне конечных точек только после выполнения интеграции, отправки данных о своих возможностях и конфигурации ее ленты.

Сообщения, передаваемые от/к коммуникационному компоненту, являются частью сценария, который происходит только с коммуникационным компонентом, и направляются непосредственно на конечную точку. Также сообщение ForwardInfo коммуникационного компонента будет направлено непосредственно на конечную точку. Лента не участвует в этой коммуникации.

Необходимо настроить способ получения сообщений, отправляемых другими конечными точками. В одном варианте сообщения могут храниться в ленте. В таком случае каждое сообщение следует запрашивать из ленты. Каждое получаемое из ленты сообщение требует подтверждения. Подтверждение получения удаляет это сообщение из ленты. В другом варианте сообщения могут получаться напрямую. В таком случае все сообщения будут получены напрямую. Если конечная точка находится не в сети, она не может получать никакие сообщения, и сообщения будут утеряны. Следовательно, конечная точка должна находиться в сети, чтобы успешно обмениваться сообщениями.

Кроме того, существует возможность, что сообщение будет сохранено в ленте конечной точки, но также будет доставлено ей напрямую. Сообщения будут доставляться напрямую в случае, если конечная точка находится в сети. Дополнительно эти сообщения будут сохранены в ленте. Когда прямая доставка сообщения подтверждается, сообщение будет автоматически удалены из ленты конечной точки. Таким образом все сообщения, которые можно получить из ленты, это сообщения, которые не были получены напрямую.

Лента клиента имеет три варианта конфигурации, только один из которых может быть активирован в любой момент времени:

- хранение сообщений в ленте;

- прямая доставка сообщений с хранением в ленте;
- прямая доставка сообщений без хранения в ленте (поведение по умолчанию).

### 6.4.3.3 Запрос сообщения из ленты

Сообщения, хранящиеся в ленте, могут быть непосредственно запрошены конечной точкой. Это показано на [Рисунке 15](#). Входящие сообщения отправляются конечной точкой А и доставляются конечной точке В. Эти сообщения хранятся в ленте конечной точки В (1). Конечная точка В отправляет запрос на сообщения из ленты на сервер через свой ящик входящих. Запрошенные сообщения изымаются из ленты и помещаются в ящик исходящих конечной точки В (2). С этого момента сообщения могут быть получены от конечной точки В (3).

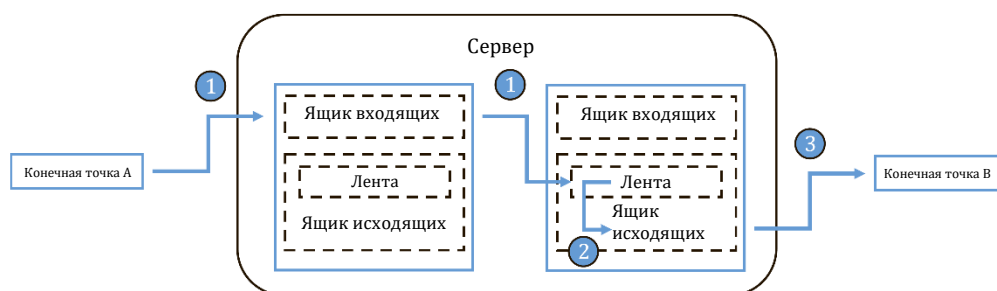


**Рисунок 15 – Визуализация получения сообщений из ленты**

Если сообщения были сброшены и удалены из ленты конечной точки, в ответном сообщении может быть указано количество сброшенных сообщений.

## 6.5 Установление подключения

Серверная система должна предоставить регистрационный код пользователю, который желает интегрировать клиентскую систему. То каким образом серверная система предоставляет регистрационный код, зависит от системы. Также серверная система решает, в какой форме предоставляется регистрационный код – числовой, цифробуквенный или иной формат, доступный к прочтению и вводу в подключаемую клиентскую систему человеком. Серверная система также может определить, будет ли ограничен срок действия регистрационного кода. Дополнительно серверная система также может определить, является ли регистрационный код одноразовым. Клиентская система должна предоставить пользователю возможность ввести регистрационный код, когда таковой используется.

Установление подключения начинается с запроса к сервису интеграции согласно Процессу интеграции (см. <https://isobus.net/isobus/efdi/#messaging.v0.m0.onboarding-process>). Это процесс предоставляет информацию для последующего обмена информацией с системой.

На основании Ответа по интеграции коммуникационный компонент должен разрешить подключение клиента только с соблюдением разрешенных критериев аутентификации и подключения.

Из сети также возможно удалить конечные точки. Такой процесс называется Процессом выведения (см. <https://isobus.net/isobus/efdi/#messaging.v0.m0.offboarding-process>). После выведения все сообщения в ленте конечной точки удаляются, а конечная точка более не имеет адреса в системе. Система также должна иметь возможность выведения на стороне сервера. Последствия такие же, как если бы конечная точка самостоятельно вышла из системы.

## 6.6 Авторизация

Авторизация для коммуникации с другими конечными точками выполняется серверной системой.

Авторизация может быть особенной в зависимости от среды и клиента. Информация об авторизации предоставляется в Процессе интеграции (см. <https://isobus.net/isobus/efdi/#messaging.v0.m0.onboarding-process>).

## 6.7 Потокковая передача данных

Коммуникационный компонент также способен осуществлять потокковую передачу. Эта способность входит в его возможности. Коммуникационный компонент без такой способности не может взаимодействовать с потоковыми данными.

Для потоковых данных следующая информация представляется важной

- Следующие глаголы
  - STREAM\_START (начать потокковую передачу)
  - STREAM\_STOP (прекратить потокковую передачу)
  - STREAM\_DATA (передавать потоковые данные)
- Описание потока/ справочная информация

Если коммуникационный компонент получает сообщение STREAM\_START с существительным, он должен отправить сообщение с глаголом STREAM\_START и запрошенным существительным всем получателям, однократно упомянутым в сообщении запроса. Когда другая конечная точка отправляет сообщение STREAM\_START с таким же существительным, это сообщение должны получить только получатели, которые не получили сообщение STREAM\_START с запрошенным существительным.

Если конечная точка получает сообщение STREAM\_START в отношении конкретного существительного, она должна начать потокковую передачу данных. Это значит, что клиент теперь должен разослать запрошенные данные, как только они станут доступны. Временной интервал для этого может зависеть от конечной точки. При получении сообщений с глаголом STREAM\_DATA, коммуникационный компонент должен переадресовать эти сообщения с существительным получателям, которые:

- а) способны получать глагол STREAM\_DATA в комбинации с существительным,
- б) подписаны на указанную выше комбинацию и
- в) запросили потоковые данные, отправив глагол STREAM\_START в комбинации с существительным.

Конечная точка может запросить прекращение потокковой передачи данных, отправив глагол STREAM\_STOP с определенным существительным. Этот глагол STREAM\_STOP должен быть отправлен открыто, если клиент более не желает получать данные. Выведение конечной точки подразумевает команду STREAM\_STOP в отношении всех потоков данных выведенной конечной точки. Если конечная точка не находится в сети, потокковая передача данных должна продолжаться, а соответствующие сообщения должны быть отправлены в ленту конечной точки-получателя (при наличии).

Когда не остается получателей данных STREAM\_DATA определенной конечной точки, коммуникационный компонент должен уведомить конечные точки, отправляющие потоковые данные, отправив сообщение STREAM\_STOP на конкретную конечную точку.

Коммуникационный компонент должен спровоцировать запрошенные конечные точки начать или прекратить отправку потоковых данных. Коммуникационный компонент должен обеспечить однократное получение запросов на начало и прекращение отправки потоковых данных конечными точками. Кроме того, коммуникационный компонент должен переадресовать входящие потоковые данные конечным точкам, которые запросили эти данные.

Данные, отправляемые с помощью глагола STREAM\_DATA, должны содержать ссылку. Эта ссылка должна вести к описанию потока. Без подобного описания потоковые данные не могут корректно интерпретироваться.

**ПРИМЕР** Когда отправляются сообщения TimeLog (хронологический журнал) согласно стандарту ISO 11783-10, соответствующий пул объектов дескрипторов устройств (DDOP) не известен, и эти сообщения TimeLog не могут быть интерпретированы. При отправке сообщений TimeLog согласно стандарту ISO 11783-10 предоставляется ссылка на DDOP. Конечная точка, которая получает такие сообщения TimeLog согласно стандарту ISO 11783-10 и соответствующую ссылку, теперь может решить, знакома ли она с такой ссылкой или еще нет. Если точка уже знакома со ссылкой, она может интерпретировать данные. Если она не знакома, она может запросить данные, на которые ведет эта ссылка.

6.8 Разделение на фрагменты

В отношении крупных данных конечные точки могут пользоваться концепцией передачи данных фрагментами. Конечные точки могут управлять такой сегментацией.

Если конечная точка желает отправить сообщение, размер которого превышает максимальный размер, указанный в MinimumCapabilities сообщения, точке необходимо разделить данное сообщение на более маленькие фрагменты. Затем конечная точка сворачивает такие фрагменты в EndpointMessages (сообщения конечной точки) и отправляет такие фрагментированные сообщения по отдельности. Конечная точка должна обеспечить правильность заполнения информации в ChunkComponent (фрагментированном компоненте) внутри сообщения. Сервер передает сообщения получателю без изменения содержимого. Получатель получает такие фрагментированные сообщения отдельными сообщениями. Затем конечная точка может заново собрать двоичное содержимое. Это показано на [рисунке 16](#).

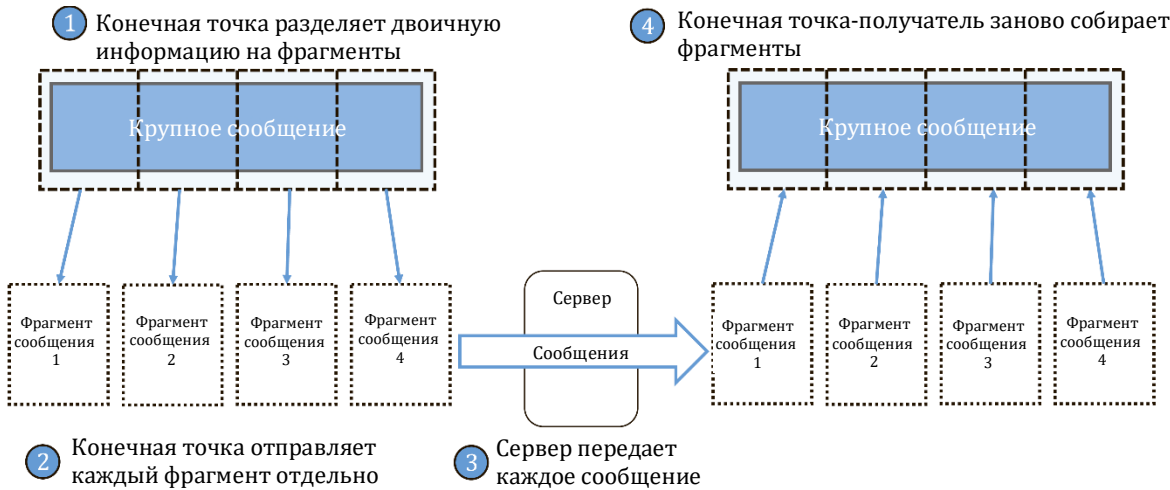


Рисунок 16 – Разделение сообщений на фрагменты

6.9 Обработка ошибок

Внутри сжатых сообщений, сообщений ServerMessages, отправляемых сервером, и сообщений EndpointMessages, отправляемых любой другой конечной точкой, задаются общие ошибки. В случае общей ошибки ответ по сценарию не содержит определенного ответного сообщения по данному сценарию. Он содержит только информацию об общей ошибке. Для каждого сценария определены дополнительные сообщения об ошибке. Причина этому заключается в том, что для каждого сценария требуются коды конкретных ошибок. Такие коды ошибок находятся в ответе по каждому сценарию и называются ResponseCode (код ответа). Каждый ответ по сценарию должен задавать возможные ошибки под ResponseCode. Технически ResponseCode представляет собой

Документы предоставляются с целью разработки национальных стандартов. Для ТК 284.

нумерование. Значение 0 всегда означает статус ОК и указывает на то, что вся обработка может быть выполнена успешно.

## 7 Формальное и семантическое описание

### 7.1 Общая информация

Перечень сообщений, сценариев, потоков выполнения сценариев и наборов сценариев является расширяемым. Эти определения общедоступны в базе данных VDMA.

Каждое сообщение, сценарий, поток выполнения сценариев и набор сценариев имеют имя пакета, состоящее из уникального идентификатора, включающего семантическое версионирование. Семантическое версионирование включает СТАРШИЕ и МЛАДШИЕ версии. Номер СТАРШЕЙ версии будет увеличен в случае публикации несовместимых изменений, а номер МЛАДШЕЙ версии будет увеличен в случае выпуска нового функционала, совместимого с существующими определениями. СТАРШИЕ версии обозначаются по схеме «v<старшая-версия>», например, «v1». МЛАДШИЕ версии обозначаются по схеме «m<младшая-версия>», например, «m4». Общая структура имеет следующую концепцию:

<уникальное-пространство-имен>.v<старшая-версия>.m<младшая-версия>.<имя>

ПРИМЕР Идентификатор основного набора сценариев обмена сообщениями:  
«messaging.v0.m0.basics»

Все файлы определений должны иметь кодировку UTF-8.

### 7.2 Существительное

Существительные являются сообщениями, задаваемыми в определениях буфера протокола. Каждое сообщение включено в пакет. Каждое сообщение имеет URL, описывающий его тип. Такой URL называется type\_url. Любой type\_url всегда выглядит следующим образом: «iso.org/5231/<имя-пакета>.<имя-сообщения>». <имя-пакета> указывается в начале каждого протокольного файла под меткой «package» (пакет), например, «messaging.v0.m0.messages». <имя-сообщения> указывается в каждом протокольном файле под меткой «message» (сообщение), например, «Onboarding».

ПРИМЕР Для сообщения об интеграции type\_url выглядит следующим образом:  
«iso.org/5231/messaging.v0.m0.messages.Onboarding».

В таком случае заполненное и сериализованное сообщение протокола буфера является данными сценария.

### 7.3 Сценарий

Все потоки выполнения сценариев описываются в файлах с расширением «scenario» (сценарий). В целом, все поля сценариев должны иметь следующую структуру: “@scenario {имя-атрибута} <значение-атрибута>”. Все поля являются обязательными. Описание каждого сценария включает:

- Идентификатор сценария
  - Описание: считываемый машиной идентификатор этого сценария, содержит семантическое версионирование, идентификатор должен быть уникальным. <идентификатор-сценария> следует заменить на идентификатор сценария.
  - Формальное представление: @scenario {идентификатор} <идентификатор-сценария>
  - Пример: @scenario {идентификатор} messaging.v0.m0.send-feed-header
- Имя сценария

- Описание: считываемое человеком имя данного сценария, без семантического версионирования. <имя-потока-выполнения-сценариев> следует заменить именем сценария.
- Формальное представление: @ scenario {имя} «<имя-сценария>»
- Пример: @scenario {имя} «Send Feed Header» (отправить заголовок ленты)
- Глагол запроса сценария
  - Описание: глагол, используемый для запроса данного сценария. <глагол-запроса-сценария> следует заменить на глагол, используемый для данного запроса.
  - Формальное представление: @ scenario {глагол.запроса} <глагол-запроса-сценария>
  - Пример: @scenario {глагол.запроса} GET
- Существительное запроса сценария
  - Описание: существительное, используемое для запроса данного сценария, определяемое type\_url используемого сообщения. <существительное-запроса-сценария> следует заменить на существительное, используемое для данного запроса.
  - Формальное представление: @ scenario {существительное.запроса} <существительное-запроса-сценария>
  - Пример: @scenario {существительное.запроса} messaging.v0.m0.messages.FeedHeader
- Глагол ответа сценария
  - Описание: глагол, используемый для ответа данного сценария. <глагол-ответа-сценария> следует заменить на глагол, используемый для данного ответа.
  - Формальное представление: @ scenario {глагол.ответа} <глагол-ответа-сценария>
  - Пример: @scenario {глагол.ответа} SHOW
- Существительное ответа сценария
  - Описание: существительное, используемое для ответа данного сценария, определяемое type\_url используемого сообщения. <существительное-ответа-сценария> следует заменить на существительное, используемое для данного ответа.
  - Формальное представление: @ scenario {существительное.ответа} <существительное-ответа-сценария>
  - @scenario {существительное.ответа} messaging.v0.m0.messages.FeedHeaderResponse

Для заполнения необходимой информации (глагола, существительного и данных) следует использовать поля для глагола и существительного конечной точки и протокольные сообщения сервера. Существительное поля представлено типом Any буфера протокола Google. Тип сообщения Any позволяет использовать сообщения как встроенные типы без их протокольного определения. Такое сообщение Any содержит поле type\_url и поле значения. Существительное должно быть вставлено в поле type\_url типа Any. Данные должны быть вставлены в поле значения типа Any. Данные являются фактически передаваемым в данном сообщении содержимым. Сценарии могут не иметь никакого содержимого.

## 7.4 Поток выполнения сценариев

Все потоки выполнения сценариев описываются в файлах с расширением «flow» (поток). В целом, все поля потоков выполнения сценариев должны иметь следующую структуру: «@flow {имя-атрибута} <атрибут>». Все поля являются обязательными. Каждый поток выполнения сценариев должен описываться следующими атрибутами:

- Идентификатор потока выполнения сценариев

- Описание: считываемый машиной идентификатор этого потока выполнения сценариев, содержит семантическое версионирование, идентификатор должен быть уникальным. <идентификатор-потока-выполнения-сценариев> следует заменить на идентификатор потока выполнения сценариев.
- Формальное представление: @flow {идентификатор} <идентификатор-потока-выполнения-сценария>
- Пример: @flow <идентификатор> messaging.v0.m0.delete-messages
- Имя потока выполнения сценариев
  - Описание: считываемое человеком имя данного потока выполнения сценариев, без семантического версионирования. <имя-потока-выполнения-сценариев> следует заменить на имя потока выполнения сценариев.
  - Формальное представление: @flow {имя} <имя-потока-выполнения-сценария>
  - Пример: @flow {имя} «Delete Messages» (удалить сообщения)
- Шаг потока выполнения сценариев
  - Описание: сценарий, который должен быть выполнен на данном шаге, идентифицируемый по его уникальному идентификатору сценария. Для опциональных сценариев идентификатор сценария указывается в квадратных скобках. Если поток выполнения сценариев включает несколько сценариев, должны присутствовать несколько подобных записей о шагах потока выполнения. Хронологический порядок сценариев определяется порядком, в котором они заданы, сверху вниз. Также различаются шаг А или шаг В. <идентификатор-сценария> следует заменить на идентификатор применяемого сценария. <направление-шага> следует заменить на «<шаг а>» для шага А или «<шаг b>» для шага В.
  - Формальное представление: @flow {направление.шага} <идентификатор-сценария>
  - Пример:
    - Опциональный сценарий: @flow {шаг.а} [messaging.v0.m0.send-feed-header]
    - Требуемый сценарий: @flow {шаг.а} messaging.v0.m0.send-feed-message-delete

## 7.5 Набор сценариев

Все наборы сценариев описываются в файлах с расширением «set» (набор). В целом, все поля набора сценариев должны иметь следующую структуру: «@set <имя-атрибута> <атрибут>». Все поля являются обязательными. Каждый набор сценариев должен описываться следующими атрибутами:

- Идентификатор набора сценариев
  - Описание: считываемый машиной идентификатор этого набора сценариев, содержит семантическое версионирование, идентификатор должен быть уникальным. <идентификатор-набора-сценариев> следует заменить на идентификатор набора сценариев.
  - Формальное представление: @set {идентификатор} <идентификатор-набора-сценариев>
  - Пример: @set <идентификатор> messaging.v0.m0.basics
- Имя набора сценариев
  - Описание: считываемое человеком имя данного набора сценариев, без семантического версионирования. <имя-набора-сценариев> следует заменить именем набора сценариев.
  - Формальное представление: @set {имя} «<имя-набора-сценариев>»

- Пример: @set <имя> «Basics» (основной)
- Поток выполнения набора сценариев
  - Описание: поток выполнения сценариев, который является частью данного набора сценариев, идентифицируемый по уникальному идентификатору сценария. <идентификатор-потока-выполнения-сценариев> следует заменить на идентификатор используемого потока выполнения сценариев.
  - Формальное представление: @set {поток} <идентификатор-потока-сценариев>
  - Пример: @set <поток> messaging.v0.m0.delete-messages

## 7.6 Структура документации

В базе данных VDMA для исходных файлов, которые содержат связанные с сообщениями и сценариями определения, имеется разграничение между сообщениями (они расположены в виде файлов буферов протокола в подпапке «messages») и определениями (они расположены в подпапке «definitions»). Каждый файл с расширением «proto» должен содержать одно определение сообщения и соответствовать схеме protobuf 3. Каждое определение сценария, потока выполнения сценариев и набора сценариев находится в папке «definitions».

Сообщения находятся по следующему адресу: <имя-пакета>/messages

Определения находятся по следующему адресу: <имя-пакета>/definitions

## 7.7 Структура файлов буферов протокола

### 7.7.1 Общая структура

Шаблон файла буфера протокола приведен на [Рисунке 17](#). Каждый файл буфера протокола должен использовать версию 3 буферов протокола. Для этого в начале файла под меткой syntax должно указываться значение = «proto3». Затем должен добавляться спецификатор пакета путем сложения метки пакета и имени самого пакета. Также для должны быть добавлены операторы импорта для использования и импортирования других определений буферов протокола. Дополнительно каждое сообщение должно иметь поле расширения. Это поле имеет размер 2048 и может использоваться для особых целей.

Каждое сообщение и каждое поле должны содержать комментарии. Для добавления документирующих комментариев для определения сообщения до начала определения сообщения следует вставить блок для комментария. Такой блок для комментария должен начинаться с наклонной черты вправо и двух звездочек. Конец блока для комментария обозначается одной звездочкой с наклонной чертой вправо. Для добавления документирующих комментариев к полям сообщения можно использовать внутрискочный комментарий после завершения определения поля. Такой комментарий должен начинаться с трех наклонных черт вправо. Конец внутрискочного блока для комментария является концом строки.

```
syntax="proto3";
package messaging.v1.m2.examples;
import "google/protobuf/any.proto";

/**
 * Комментарий к примеру сообщения.
 */
message ExampleMessage {

  string endpoint_id=1;                /// Это комментарий к полю

  repeated google.protobuf.Any extension = 2048; /// Для особых целей
}
```

Рисунок 17 – Общая структура файлов буфера протокола

### 7.7.2 Структура кода ошибки

Каждое сообщение, используемое как существительное ответа в сценарии, должно содержать код ответа. Структура одинакова для всех сообщений и показана на шаблоне, [Рисунок 18](#). Для этого в каждое сообщение включена нумерация с именем ResponseCode (код ответа). Поле должно называться «response\_code» и использовать индекс поля «1». Такая нумерация содержит специальную информацию о сообщении и указывает на успешное выполнение обработки или наличие ошибок. Запись со значением 0 всегда означает статус ОК. Это случай по умолчанию. И наоборот, это означает, что если ResponseCode не равен 0, то что-то не было выполнено со статусом ОК. Поле с числом 1 имеет имя «response\_code» и тип «ResponseCode».

```
syntax="proto3";
package messaging.v1.m2.examples;
import "google/protobuf/any.proto";

/**
 * Комментарий к сообщению для существительного ответа шаблона сценария.
 */

message TemplateScenarioResponseNoun {

  enum Responsecode {

    OK = 0;                                     ///Нет ошибки

  }

  ResponseCode response_code = 1;               /// Содержит информацию об обработке

  repeated google.protobuf.Any extension = 2048; /// Для особых целей

}
```

Рисунок 18 – Шаблон буфера протокола ответа сценария

## 8 Структурирование транспортного уровня

### 8.1 Общая информация

Сообщения, передаваемые коммуникационным компонентом, имеют формат, описанный внутри ServerMessage (серверное сообщение). Сообщения, отправляемые коммуникационному компоненту, имеют формат, описанный внутри EndpointMessage (сообщение конечной точки).

Этот протокол технически не привязан к MQTT. В данной редакции стандарта только MQTT должен использоваться в качестве транспортного протокола, поэтому в пунктах далее будет описано применение протокола MQTT.

На [Рисунке 19](#) изображена коммуникация с сервером через MQTT. В таком случае конечная точка опубликует сериализованные буфером протокола сообщения EndpointMessages в теме «ящик входящих», а сервер подпишется на эту тему. Сообщения, отправляемые на конечную точку, будут публиковаться сервером в качестве сериализованных буфером протокола сообщений ServerMessage в теме «ящик исходящих». Конечная точка должна подписаться на эту тему. Обе темы указываются в ответе после интеграции. Опрос не требуется.

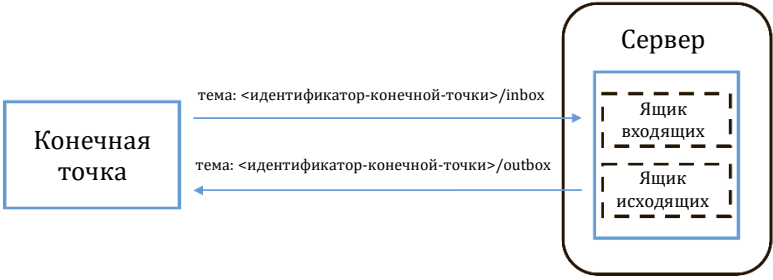


Рисунок 19 – Коммуникация с сервером через MQTT

Качество MQTT на уровне сервиса для публикаций и подписок должно быть равно 2.

8.2 Обнаружение сервиса

8.2.1 Общая информация

Через протокол MQTT МИДС должен быть объявлен под именем сервиса «efdi-onboarding». В зависимости от объема использования сети Интернет обнаружение сервиса должно применяться с использованием DNS-SD [DNS-SD] для подключения интернет-систем. В случае локальных сетей для обнаружения доступного сервиса интеграции должен применяться механизм mDNS [MDNS].

8.2.2 Подключение через Интернет

Сервис интеграции МИДС использует такой же механизм просмотра DNS-SRV-Lookup, как и описанный в разделе 3.2.1 протокола [XMPP]. Предпочтительный способ присваивания FQDN заключается в использовании записей [DNS-SRV] следующим образом:

- а) Иницирующий объект создает запрос DNS-SRV, в котором есть следующие вводные данные:
  - 1) сервис «интеграции МИДС»
  - 2) протокол «TCP»
  - 3) имя, соответствующее «исходному домену» TLS-CERTS [TLS] сервиса обмена данными ИСУФ, к которому иницирующий объект желает подключиться (например, «example.net» или «flow.example.com»)
- б) Результат является запросом, например, \_efdi-onboarding\_tcp.example.net.  
или  
\_efdi-onboarding\_tcp.flow.example.com.
- в) Если ответ получен, он будет содержать одну или более комбинации указаний на порт и FQDN, каждое из которых взвешивается и располагается в соответствии с приоритетом согласно DNS-SRV. (Однако если результатом просмотра DNS-SRV является единственная запись ресурса с целевым адресом «.», т. е. корневым доменом, то иницирующий объект должен сразу остановить обработку SRV, так как согласно DNS-SRV такой целевой адрес означает, что сервис определенно недоступен в этом домене.
- г) Иницирующий объект выбирает по меньшей мере один из полученных для присваивания FQDN (согласно правилам DNS-SRV), для чего он выполняет просмотры DNS A или AAAA по этому FQDN; результатом является адрес IPv4 или IPv6.
- д) Иницирующий объект использует этот IP адрес(-а) из успешного присвоенного FQDN (с соответствующим номером порта, полученным после просмотра DNS-SRV) в качестве адреса подключения для получающего объекта.

- е) Если иницирующий объект не может подключиться, используя такой IP адрес, но просмотры А или АААА выдают в качестве результата несколько IP адресов, то иницирующий объект использует следующий присвоенный IP адрес для такого FQDN в качестве адреса подключения.
- ж) Если иницирующий объект не может подключиться и использовал все присвоенные IP адреса для данного FQDN, тогда он повторяет процесс присваивания и подключения для следующего FQDN, полученного после просмотра DNS-SRV, в соответствии с приоритетом и весом согласно определению DNS-SRV.
- з) Если иницирующий объект получает ответ на свой запрос DNS-SRV, но не может установить связь с сервисом интеграции, используя полученные в ответе данные, то ему ЗАПРЕЩЕНО пробовать выполнять процедуру возврата в исходный режим, описанный в следующем разделе (таким образом предотвращается несоответствие статусов входящих и исходящих соединений).
- и) Если иницирующий объект не получает ответ на свой запрос DNS-SRV, он ОБЯЗАТЕЛЬНО должен попробовать выполнить процедуру возврата в исходный режим, описанную в следующем разделе.

Если обнаружение сервиса безуспешно, система подключения должна обеспечить способ ручной конфигурации необходимой информации о подключении.

### 8.2.3 Локальная сеть

Аналогично определению записей DNS для подключения через Интернет способ обнаружения сервиса mDNS должен пользоваться этими же записями.

Приложение А  
(справочное)

Глаголы OAGIS

Компания Open Applications Group, Inc. (OAGi) создала Спецификацию Open Applications Group по интеграции (OAGIS), в которой перечислены глаголы и содержится шаблон сообщений/ответов с глаголами, используемые для описания сценариев. Перечень глаголов OAGIS включает следующие глаголы:

- Acknowledge (квитировать)
- Cancel (отменить)
- CancelAcknowledge (отменить квитирование)
- Change (изменить)
- ChangeAcknowledge (изменить квитирование)
- Confirm (подтвердить)
- Get (получить)
- Notify (уведомить)
- Process (обработать)
- Show (показать)
- Sync (синхронизировать)
- SyncResponse (синхронизировать ответ)

Пример и шаблон сообщения/ответа даны в [Таблице А.1](#).

Таблица А.1 — Шаблон сообщения/ответа OAGIS

| Глагол сообщения                                                    | Глагол ответа      | Пример                                                                                                                                            |
|---------------------------------------------------------------------|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| Process                                                             | Acknowledge        | Запрос: ProcessPurchaseOrder (обработать заказ на закупку)<br>Ответ: AcknowledgePurchaseOrder (квитировать заказ на закупку)                      |
| Change<br>(с поддержкой добавления, изменения, удаления или замены) | ChangeAcknowledge  | Запрос: ChangePurchaseOrder (изменить заказ на закупку)<br>Ответ: ChangeAcknowledgePurchaseOrder (изменить квитирование заказа на закупку)        |
| Cancel                                                              | CancelAcknowledge  | Запрос: CancelPurchaseOrder (отменить заказ на закупку)<br>Ответ: CancelAcknowledgePurchaseOrder (отменить квитирование заказа на закупку)        |
| Sync<br>(с поддержкой добавления, изменения, удаления или замены)   | SyncResponse       | Запрос: SyncCustomerAddress (синхронизировать адрес заказчика)<br>Ответ: SyncResponseCustomerAddress (синхронизировать ответ с адресом заказчика) |
| Get                                                                 | Show               | Запрос: GetPurchaseOrder (получить заказ на закупку)<br>Ответ: ShowPurchaseOrder (показать заказ на закупку)                                      |
| Notify                                                              | Ответ не требуется | NotifyRemittanceAdvice (уведомить об оплате)                                                                                                      |

Использование глаголов OAGIS описано в пункте [6.2](#). Иерархическая структура наборов сценариев, потоков выполнения сценариев и сценариев. Используемые и расширенные определения приведены в пункте [6.2.2](#) «Сценарий».

## Приложение В (справочное)

### Перевод элементов ISO 11783-10 в буфер протокола

#### В.1 Общий подход

##### В.1.1 Общее описание

Присвоение порядковых номеров, представленное в данном руководстве, во многом является толкованием определений структуры XML стандарте ISO 11783-10:2015 на язык описания сообщений в версии 3 буфера протокола.

##### В.1.2 Соглашение о наименованиях

Соглашение о наименованиях полей в сообщениях буфера протокола следует правилам, изложенным документации к протоколу. Имена соответствуют именам элементов и атрибутов, указанных в стандарте ISO 11783-10:2015, Приложении D в соответствующих колонках «Атрибуты» таблиц описания элементов. Однако отличие заключается в том, что они пишутся слитно с Заглавной Буквы в случае имен сообщений, а в случае имен полей пишутся через знак подчеркивания согласно требованиям вышеупомянутого стандарта оформления. Для имен элементов следует всегда использовать полные слова (например, Time (время) вместо TIM, Device (устройство) вместо DVC). Текстовое описание перечисляемых значений берется из содержимого элемента `xsd:definition` соответствующего определения перечисляемого значения в файлах XML Schema для файлов задач в формате ISOXML, расположенных здесь: <http://dictionary.isobus.net/isobus/file/supportingDocuments>. Пробелы заменяются знаком подчеркивания, а скобки опускаются. Впереди добавляется имя с заглавной буквы соответствующего атрибута из колонки XML таблиц стандарта ISO 11783-10:2015, Приложения D плюс знак подчеркивания (например, C\_SETTABLE, A\_NOTSET и пр.). Такой подход необходим для обеспечения необходимого ограничения уникальности диапазона имен сообщений/пакетов в перечислениях буферов протокола.

##### В.1.3 Последовательность и номера полей

Последовательность полей соответствует последовательности, заданной алфавитным порядком атрибутов согласно стандарту ISO 11783-10:2015. Номера полей и сообщений, стандартизированных в качестве элементов и/или атрибутов в стандарте ISO 11783-10:2015, назначаются в диапазоне 0–2047 (см. <https://developers.google.com/protocol-buffers/docs/proto3#assigning-tags>, так как номера свыше 2047 больше не смогут кодировать varint в пределах размера в 2 байта). Никакой номер в этом диапазоне не должен использоваться в качестве отдельных или особых расширений. Расширения будут иметь номера полей от 2048 и выше.

##### В.1.4 Структурирование типов данных

Применяется следующее структурирование типов данных. К типу XML Schema относится содержимое колонок «Тип» в таблицах определения элементов в стандарте ISO 11783-10:2015, Приложении В. См. [Таблицу В.1](#).

**Таблица В.1 — Структурирование типа XML Schema относительно типа Protobuf**

| Тип XML Schema  | Тип Protobuf           |
|-----------------|------------------------|
| xs:double       | число двойной точности |
| xs:unsignedLong | uint64                 |

| Тип XML Schema                                                                                                                               | Тип Protobuf                                                                                            |
|----------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------|
| xs:unsignedByte                                                                                                                              | uint32                                                                                                  |
| xs:unsignedShort                                                                                                                             |                                                                                                         |
| xs:long                                                                                                                                      | sint64                                                                                                  |
| xs:decimal                                                                                                                                   | число двойной точности                                                                                  |
| xs:hexBinary при использовании с ограничением по длине 2 для номеров DDI                                                                     | uint32                                                                                                  |
| xs:hexBinary                                                                                                                                 | байты                                                                                                   |
| xs:string                                                                                                                                    | Строка                                                                                                  |
| xs:ID при использовании для внутренних ссылок в файле задачи в формате ISOXML (шаблон из трех букв + число, например, (GGP GGP-)([0-9])+)    | UID (см. также iso11783p10.v1.m0.messages)                                                              |
| xs:IDREF при использовании для внутренних ссылок в файле задачи в формате ISOXML (шаблон из трех букв + число, например, (GGP GGP-)([0-9])+) |                                                                                                         |
| xs:ID при использовании для имен сопроводительных файлов оригинального набора данных TASKDATA.XML                                            | строка                                                                                                  |
| xs:NMTOKEN при использовании для перечисления файлов ISOXML                                                                                  | Перечислимый тип. Для получения более подробной информации о перечислении также см. <a href="#">B.3</a> |
| xs:dateTime                                                                                                                                  | См. подробное описание представления временной метки в пункте <a href="#">B.5</a> .                     |
| Другие способы представления времени и/или даты (например, в виде целых чисел как в двоичных хронологических журналах).                      |                                                                                                         |

Такие ограничения, как шаблоны, максимальные значения и пр. согласно стандарту ISO 11783-10:2015 не применяются в качестве части формата присвоения порядковых номеров буферов протокола (и не являются необходимыми для передачи данных через проводную связь). При необходимости от должны проверяться на принимающей стороне.

## B.2 Значения по умолчанию

Protobuf 3 имеет специальный процесс инициализации структуры сообщений для дальнейшей обработки после получения сообщений, который строго соответствует подходу статической типизации и обеспечивает присваивание четко определенных значений каждой переменной. Концепция полей, указывающих на возможность нулевого значения, или содержимого NULL (неопределенное значение) как в базах данных отсутствует в буфере протокола.

В местах, где должно быть передано понятие «пустого» или «неопределенного» (по сравнению с 0) поля, следует использовать сжатое сообщение (заданное в [google/protobuf/wrappers.proto](#)). Для двойных полей согласно IEEE 754 может использоваться одно из NaN (нечисловое значение). Для полей со строками эта проблема не является актуальной, так как большинство таких полей представляют собой задаваемые пользователем кодированные данные, а пустая строка не несет никакого значения.

См. информацию о значениях по умолчанию в перечислениях в пункте [B.3](#).

## B.3 Перечисления

Все перечисления в протоколе ISOBUS реализуются как анонимные типы, встроенные в соответствующие элементы, а потому практически не используемые повторно и привязанные лишь к одному элементу. По этой причине перечисления на основе определений элементов

согласно ISO 11783-10:2015 также должны быть заявлены как типы, встроенные в соответствующие сообщения, в определениях буферов протокола.

Согласно языку описания proto3 перечисления должны включать значение 0 enum (см. <https://developers.google.com/protocol-buffers/docs/proto3#enum>). Практически все перечисления в ISO 11783-10:2015 начинаются со значения 1. Поэтому для структурирования буфера протокола добавляется числовое значение 0 с текстовым описанием NOT\_SET (не задано).

Исключение из этого правила может быть применимо к полям, определенным как опциональные в элементах ISO 11783-10:2015. Для полей с заданным значением 0, которое имеет другой смысл – не отсутствующее значение (неопределенное значение), применяются следующие исключительные определения для следующих полей элементов

- GPN (GuidancePattern) Attribute N (NumberOfSwathsLeft) – шаблон движения, количество полос слева
- GPN (GuidancePattern) Attribute O (NumberOfSwathsRight) – шаблон движения, количество полос справа

Для этих исключений атрибуты сообщения буфера протокола не будет определено как скалярный значимый тип, как это описано в пункте [B.1.4](#). Вместо этого будут использоваться соответствующие сжатые сообщения в качестве типа атрибута (определены в google/protobuf/wrappers.proto). Таким образом можно разграничить значение 0 и неопределенное значение.

Сжатые сообщения не используются для всех прочих атрибутов сообщений для сокращения потребления ресурсов, возникающего при определении атрибута сообщением.

## **В.4 Идентификаторы**

### **В.4.1 Уникальные идентификаторы согласно ISO 11783-10:2015, 8.4**

Уникальные идентификаторы для объектов согласно описанию в стандарте ISO 11783-10:2015 8.4 должны быть перекодированы в буферы протокола с помощью типа сообщений UID. Эти трехбуквенные коды элементов должны быть выделены из идентификатора. Они не кодируются в буфере протокола. Такой подход исключает двусмысленность, так как каждый элемент типа xs:ID/xs:IDREF в ISO 11783-10:2015 идентифицирует/указывает только на один элемент. Трехбуквенные коды предназначены только для гарантированной уникальности идентификаторов в пределах файла taskdata.xml. Единственными существенными исключениями, когда в файле ISOXML могут присутствовать ссылки на различные элементы, являются атрибут A (filename – имя файла) элемента XFR и атрибут A (ObjectIDRef – ссылка на идентификатор объекта) элемента LNK. Первый кодирует имя файла, которое согласно пункту [5.3](#) данного руководства должно быть перекодировано в виде строки в буфер протокола. А для второго определение буфера протокола использует механизм прямой интеграции идентификаторов списка ссылок LinkList, если они могут быть представлены в виде URI, в сообщения буфера протокола. Этот подход описан более подробно далее в этом разделе.

Числовое значение после трехбуквенных кодов кодируется в числовом поле сообщения UID. Хотя эта схема допускает наличие конечных нулей в определении шаблона, они не допускаются стандартом (см. ISO 11783-10:2015, 8.4). Поэтому безопасно кодировать числовую часть в виде sint64.

### **В.4.2 Использование унифицированных идентификаторов ресурса**

Сообщение UID основано на синтаксисе URI в глобально уникальной идентификации согласно описанию в RFC 3986.

Для кодировки URI используется поле uri в сообщении UID. Таким образом URI могут быть заданы для объекта, который содержит информацию об идентификации. Одному объекту возможно

назначить несколько URI. Это может оказаться полезным, например, если есть потребность назначить номер изделия и произвольный UUID, генерируемый приложением. Для конвертации UUID в URN следует использовать префикс пространства имен urn:uuid:. К другим важным предварительно заданным префиксам пространства имен относятся: iso11783:pid: для идентификатора изделия согласно ISO 11783-12 и iso11783:cn: для ClientNAME (имя клиента) согласно ISO 11783-5:2019 и ISO 11783-10:2015.

### В.4.2.1 Идентификация устройства

Сообщение UID содержит многочисленные идентификаторы. Таким образом сообщения элементов могут указывать несколько идентификаторов для одного объекта. Кроме того тип идентификаторов не ограничен конкретным типом идентификации, например NAME, VIN или UUID. Схема идентификаторов представляет собой унифицированное имя ресурса (URN), которое может быть указано для каждого типа идентификатора.

В определении элементов устройств согласно ISO 11783-10 применяется приоритетный список типов идентификации:

- «Идентификация изделия» (PI) согласно ISO 11783-12:2019
- ИМЯ ISOBUS согласно ISO 11783-10:2015 (задается как глобально уникальное)
- UUID версия 4 согласно RFC 4122 при наличии конечной точки без какого-либо прошлого идентификатора

### В.4.2.2 Обнаружение идентификации устройства

Обнаружение идентификации устройства следует правилу, определенному в руководстве AEF TC-CORE.

Для получения полей PI согласно ISO 11783-10:2015 TC или DL используют связь согласно ISO 11783-12:2019:

- A.14 Код идентификации изделия
- A.15 Бренд идентификации изделия
- A.16 Модель идентификации изделия

Таким образом нет необходимости изменения взаимодействия между TC или DL с клиентом TC или клиентом DL согласно ISO 11783-10:2015.

Применимы следующие правила:

- Каждый раз, когда клиент TC или DL активирует свой DDOP через TC или DL, TC или DL должен проверить наличие соответствующей ссылки, содержащей информацию по идентификации изделия.
- Таким образом TC или DL необходимо получить информацию по идентификации изделия. Они могут запросить PGN 0xFC8D от всех ЭБУ или конкретного ЭБУ в сети, используя собственный адрес отправителя в любой момент, или получить информацию от диагностического блока, встроенного в тот же аппаратный компонент, в который встроены TC или DL.
- TC или DL должны добавить идентификацию изделия в файлы передачи данных, когда клиент активирует свой DDOP.
  - При отсутствии приложенного файла типа «LINKLIST» (перечень ссылок) TC или DL должны его создать, а если такой файл уже существует, то TC или DL должны добавить информацию в имеющийся файл.
  - При наличии ссылки на соответствующее изделие TC или DL должны обновить эту ссылку, используя текущую идентификацию изделия.

- При отсутствии ссылки для подключенного изделия TC или DL должны создать новую ссылку.
- TC или DL не должны использовать идентификацию изделия по ссылке для проверки соответствия хранящегося DDOP подключенному изделию (для такой проверки применяются только правила ISO 11783-10:2015).
- Если клиент не предоставляет информацию по идентификации изделия, для такого клиента не нужно добавлять ссылку в прикрепляемый файл.

Программное обеспечение ИСУФ может добавлять ссылки с идентификацией изделия для каждого DDOP, на который дается ссылка в TASKDATA (данные о задаче), при записи файлов обмена данными.

## V.5 Временные метки

В любом месте, где необходимо полностью передать информацию о времени вместе с датой или дату отдельно, применяются однозначные временные метки UTC. Если элемент ISO 11783-10:2015 содержит отдельные поля даты и времени, которые семантически связаны друг с другом, они объединяются в одно поле типа временной метки буфера протокола (например, см. элемент PTN (позиция)). Точное низкоуровневое представление соответствующего типа данных буфера протокола и примеры кода даны здесь: <https://developers.google.com/protocol-buffers/docs/reference/google.protobuf.Timestamp>.

Полные временные метки с информацией о часовом поясе были введены в версии 4 стандарта ISO 11783-10:2015. При создании сериализации буферов протокола на основе файлов ISOXML можно пользоваться этой информацией.

## V.6 Хронологический журнал в формате XML и в двоичном формате

Определение сообщения буфера протокола с хронологическим журналом следует оригинальному определению XML согласно ISO 11783-10:2015 с одним заметным отличием: вводится повторяемое сообщение времени. При конвертации из набора данных задачи атрибуты имени и длины файла должны быть заполнены согласно сообщению буфера протокола. Затем содержимое файла в двоичном формате должно быть конвертировано – одно сообщение времени на одну запись – в набор сообщений времени. Этот процесс также включает конвертацию времени, указанного целыми числами, и координат, данных в файле двоичного формата, во временную метку и данные с плавающей запятой соответственно. Кроме того, любые значения по умолчанию, заданные в файле TLG\*.XML, должны добавляться непосредственно в виде DataLogValues (значения журнала данных). Такие сообщения времени затем должны добавляться в виде повторяемых подсообщений к сообщению хронологического журнала. Описание структуры сопроводительного файла TLG\*.XML затем может быть безопасно удалено, так как оно может быть легко восстановлено при обратной конвертации. Такой подход обеспечивает унификацию двоичного формата и формата XML только для представления на основе одного буфера протокола.

Для обратной конвертации необходимо следовать процедуре выше в обратном порядке, задать имена файлов в соответствии с содержимым полей имени файлов и восстановить файл TLG\*.XML. Прямая и обратная конвертация происходит без потерь за исключением незначительных потерь точности в связи с конвертацией между целыми числами и числами с плавающей запятой, а также порядок значений DLV в восстановленных двоичных файлах может отличаться от оригинального набора данных. Дополнительно можно определить значения по умолчанию для DDI и поместить их обратно в файл TLG\*.XML, так как они должны содержать то же значение.

Такие же сообщения времени, как описано выше, могут использоваться для постоянной потоковой передачи регистрационных данных.

## Библиография

- [1] ISO 11783-5, *Тракторы и машины для сельского и лесного хозяйства. Сеть данных последовательного управления и коммуникации. Часть 5: Управление сетями*
- [2] ISO 11783-10:2015, *Тракторы и машины для сельского и лесного хозяйства. Сеть данных последовательного управления и коммуникации. Часть 10: Обмен данными контроллера задач и информационной системы менеджмента*
- [3] ISO 11783-12:2019, *Тракторы и машины для сельского и лесного хозяйства. Сеть данных последовательного управления и коммуникации. Часть 12: Диагностические сервисы*
- [4] IETF RFC2782, *DNS RR для указания расположения сервисов (DNS SRV)*
- [5] IETF RFC6763, *Обнаружение сервисов на основе DNS (DNS-SD)*
- [6] IETF RFC6762, *Многоадресный DNS (MDNS).*
- [7] IETF RFC5246, *Протокол защиты транспортного уровня (TLS)*
- [8] IETF RFC6120, *Расширяемый протокол обмена сообщениями и информацией о присутствии (XMPP): Основы*
- [9] OASIS [mqtt-v3.1.1], *MQTT, версия 3.1.1*
- [10] *proto3, Версия буферов протокола 3*



Документы предоставляются с целью разработки национальных стандартов. Для ТК 284.