

ООО «ГТЛАБ-Диагностика»

## **GTLD-SCADA-PUBLISHER**

**Общий интерфейс публикации данных**

Руководство системного программиста

Саров 2025

## Основные сведения

Для публикации данных в **GTLD-SCADA** предусмотрен HTTP-интерфейс, функционирование которого обеспечивается сервисом **GTLD-SCADA-PUBLISHER**.

Данные передаются посредством HTTP-POST запросов в формате JSON.

Различается два механизма использования интерфейса:

- передача состояний;
- передача событий.

Далее в руководстве раскрыты эти механизмы.

## 1. Передача состояний

Под состоянием понимается значение переменной, измеренное либо полученное математическим путём в текущий момент времени, которое характеризует каким-либо образом состояние объекта диагностики.

Внешняя система осуществляет периодическую отправку состояний в интерфейс **GTLD-SCADA-PUBLISHER**, после чего данные могут быть обработаны в системе: сохранены в статистике, отображены пользователю и т.п.

На рисунке 1 представлен общий формат передачи данных.

```
let __data = [  
  {  
    route_key: "Маршрутный ключ",  
    value: 123,  
  },  
];
```

Рисунок 1 – Формат передачи данных

- поле **route\_key** является идентификатором данных (маршрутный ключ) внутри **GTLD-SCADA**, все обращения к данным внутри системы, хранение статистики, доступ визуализаторов к данным и т.п. выполняется по полю **route\_key**;
- поле **value** содержит значение которое подвергается публикации. Поле value может быть строкой, числом с плавающей точкой, а также объектом с произвольной структурой.

**Замечание:** Обращаю внимание на то, что в примере переменная **\_\_data** определена, как массив, т.е. интерфейс **GTLD-SCADA-PUBLISHER** ожидает получить JSON-массив объектов, соответственно, если в массиве одно значение, публикация будет выполнена для одной переменной, но возможна публикация значений сразу нескольких переменных.

Более сложный пример представлен на рисунке 2.

```
let __data = [  
  {  
    route_key: "rmsA1",  
    value: 12.3,  
  },  
  {  
    route_key: "signalState",  
    value: {  
      iepe: true,  
      adcOnline: true,  
      ip: "192.168.1.1",  
      connectionsCount: 5,  
    },  
  },  
];
```

Рисунок 2 – Пример публикации нескольких переменных

При создании пользовательского источника данных, для него может быть определён параметр **TTL** (time to live – время жизни). В этом случае все механизмы визуализации будут его учитывать и отмечать неактуальные данные, если они не обновлялись дольше чем указано в данном параметре.

Создание пользовательских источников данных и прикрепление их к различным визуализаторам осуществляется в WEB-интерфейсе **GTLD-SCADA** и отражено в соответствующем руководстве пользователя.

## 2. Передача событий

Передача событий выполняется таким же способом, как передача данных, отличием является то, что формат поля **value** строго регламентирован, а **route\_key** всегда должен иметь значение зарезервированного системного источника данных: **scada\_system\_journal**.

На рисунке 3 представлен полный пример объекта, передаваемого в интерфейс **publisher** для фиксации события.

```
let __data = [
  {
    route_key: "scada_system_journal",
    value: {
      mod_id: "streamer_1",
      level: "CRITICAL",
      code: 2002,
      text: "Потеряна связь с АЦП",
      tag: "STREAMER",
      trigger: "SET",
    },
  },
];
```

Рисунок 3 – Пример структуры объекта для публикации события

В таблице 1 представлен формат объекта события для публикации в GTLD-SCADA.

Таблица 1 – Формат объекта события

Ключ	Наименование	Комментарий
mod_id	Идентификатор источника	Каждый модуль и подсистема должны иметь свой уникальный идентификатор, для модулей STREAMER и GTLD-DAEMON эти идентификаторы указываются в полях modID конфигурационных файлов.  Пустая строка не допустима.
level	Уровень важности	Допустимо 3 уровня важности: <ul style="list-style-type: none"><li>• CRITICAL</li><li>• WARNING</li><li>• INFO</li></ul> По умолчанию: INFO

Продолжение таблицы 1

Ключ	Наименование	Комментарий
code	Код события	<p>Цифровой код события, классификация событий представлена в соответствующем руководстве. Следует ознакомиться с классификацией прежде чем использовать это ключ.</p> <p>По умолчанию: 0</p>
text	Текстовое описание события	<p>Формировать это поле следует таким образом, чтобы избежать упоминание информации, содержащейся в остальных ключах</p>
tag	Тэг события	<p>Используется для классификации событий по группам, могут быть произвольными, но с учетом занятых наименований:</p> <ul style="list-style-type: none"> <li>• COMMON;</li> <li>• SECURITY;</li> <li>• STREAMER;</li> <li>• DAEMON;</li> <li>• CONFIG;</li> <li>• ACTIONS.</li> </ul> <p>По умолчанию: COMMON</p>
trigger	Триггер состояния	<p>Данный ключ следует использовать в том случае, если событие свидетельствует о переходе в некоторое состояние, которое будет отменено аналогичным последующим событием. Например, потеря связи с АЦП (последующее появление связи с АЦП отменит это состояние), переход порта АЦП в недопустимое состояние (последующий переход в нормальное состояние порта отменит данное событие), превышение какого-либо контролируемого параметра и т.п.</p> <p>Предусмотренные значения:</p> <ul style="list-style-type: none"> <li>• SET;</li> <li>• UNSET;</li> <li>• UNDEFINED.</li> </ul> <p>По умолчанию: UNDEFINED</p>