文档编号：

版 本 号：

**GODAAS振动采集跨平台版本开发方案**

编 写 人：殷伟文

审 核 人：

**2025年1月**

**修订记录**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **序号** | **修改状态** | **修改页码及条款** | **修改人** | **审核人** | **修改生效日期** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

注：

修改状态为 A－增加，D－删除，M－修改

对该文件内容增加、删除或修改均需填写此变更记录，详细记载变更信息，以保证其可追溯性。

提交用户时，将此变更记录删除

目 录

[1. 引言 - 1 -](#_Toc17227)

[1.1. 文档目的 - 1 -](#_Toc20648)

[1.2. 主要参考资料 - 1 -](#_Toc16647)

[1.3. 术语定义 - 1 -](#_Toc17013)

[2. 说明 - 1 -](#_Toc12980)

[2.1. 需求规定 - 1 -](#_Toc32720)

# 引言

## 文档目的

本文档主要用于描述GODAAS振动采集跨平台版本开发方案，便于公司测试、分析人员对数据处理流程的理解和验证。

## 主要参考资料

1. 《计算机软件文档编制规范》(GB/T8567-2006)；

## 术语定义

表 1.4‑1 术语定义表

|  |  |
| --- | --- |
| **术语名称** | **说明** |
|  |  |
|  |  |

# 数据流程

## 数据处理流程



1. TCP 服务

负责监听和接收振动设备传输过来的原始数据，作为整个数据处理流程的入口。

TCP Listener：用于建立与设备的TCP连接并接收数据。

1. 协议适配器

对接收的原始数据进行协议解析和处理，确保数据格式与系统的后续处理模块兼容。包括:

配置读写：根据设备和系统的配置进行数据的读写操作。

心跳：处理设备的心跳包以检测设备的在线状态。

数据：对振动数据进行解析和初步处理。

示波启停：控制振动数据的采集和停止。

1. 控制器

负责核心数据的处理、分析和管理。主要有如下几部分组成：

示波器：用于实时显示振动波形数据。

采样控制：控制振动数据的采样频率和采集方式。

计算：对振动数据进行计算和分析（如频率、幅值等）。

存储/转发：将处理后的数据存储到数据库或转发到其他系统。

1. UI（用户界面）

通过HTTP或WebSocket（WS）接口与控制器通信，向用户展示振动数据及系统信息，并提供相关的配置操作。

列表展示：展示设备列表或数据列表。

通道设置：用于配置振动采集的通道参数。

波形预览：实时显示振动数据的波形图。

系统参数：显示系统运行参数和状态。

在技术选型上，跨平台解决方案中，我们选用**Golang**作为主要开发语言。并且通过Go-template技术通过HTML实现前端展示。并且通过WS实现实时数据刷新。Golang程序开发支持高并发TCP连接以及协议数据处理，并考虑项目应用场景，增加数据二次计算（如索力计算、公式计算）以及数据推送的功能。

在边缘网关中，已经实现了部分振动采集的功能。当前实现的功能架构如下图所示：



# 功能对齐

在边缘网关中实现了振动采集的部分功能。对比DAAS软件功能，列出如下清单（见下表）。

* DAAS分析功能不在此表中
* UI展示相关功能优先级低
* 以下打√为已经实现。打X为优先级较低，暂不实现

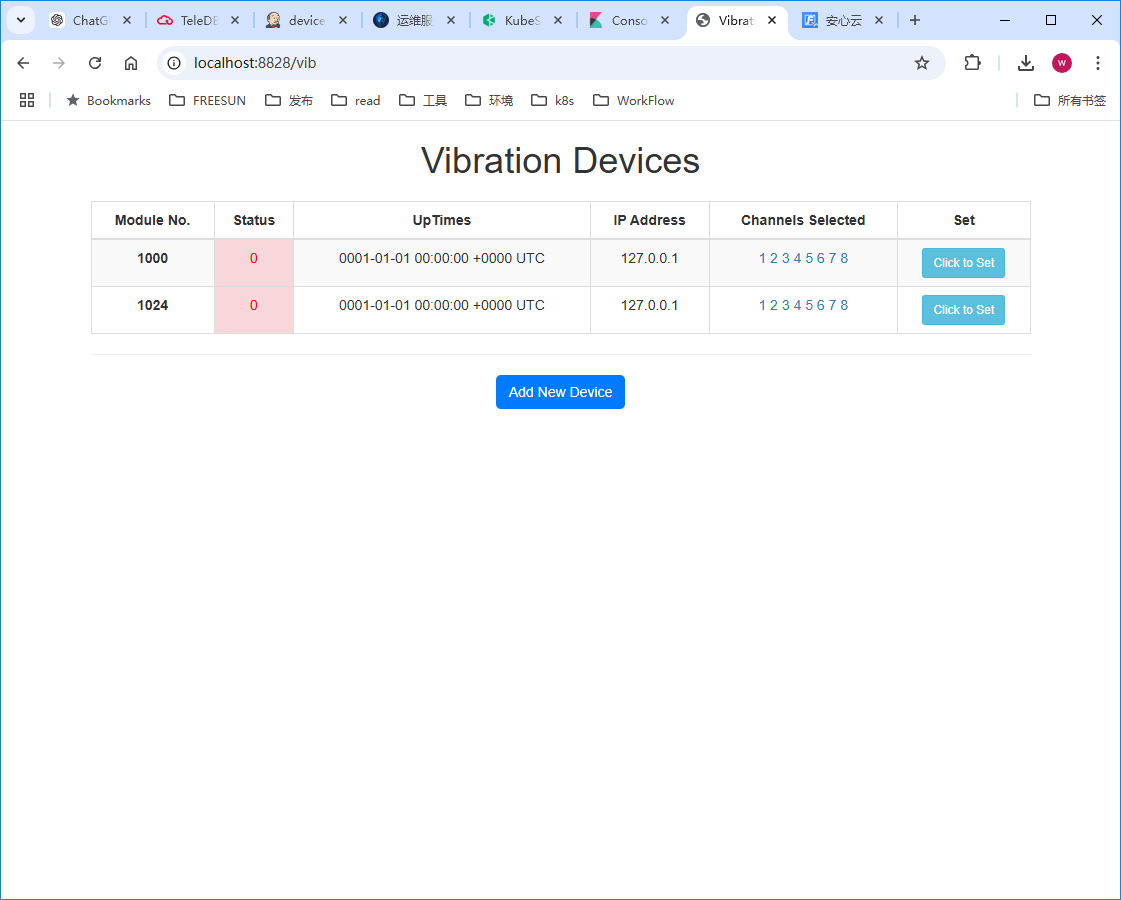
|  |  |  |  |
| --- | --- | --- | --- |
| **功能模块** | **功能点** | **GoDAAS** | **备注** |
| 采样示波 | 示波（实时、原始电压、时域图） | √ | 不支持复杂显示（时频同显、利萨如、瀑布）、不支持显示点数修改、不支持物理量显示； 回放：选择历史数据文件展示 |
| 示波（实时、原始电压、频域图） | √ |  |
| 采样触发-定时 | √ |  |
| 采样触发-连续 |  |  |
| 采样触发-通道触发 | √ |  |
| 采样触发-手动 |  |  |
| 回放 | X | 显示相关，优先级低 |
| 采样文件保存方式 |  | 目前GoDAAS存入influxdb，修改支持文件ODB格式 |
| 去直流 |  |  |
| 参数配置 | 基础通道参数配置 | √ |  |
| 硬件触发方式 |  |  |
| GPS同步模式设置 |  |  |
| 自动校准设置 |  |  |
| 触发采样-滞后点数 |  |  |
| 定时采样-采样次数 |  |  |
| 定时采样-连续采集 |  |  |
| 时钟采样 | X | 使用场景少，暂不实现 |
| 设备详情 | 在线设备列表 |  | 可以查看当前连接的设备，以及每个设备当前的通道配置。 详情中可以查看（用于DEBUG）： 设备编号：设备的唯一标识 IP地址：设备的当前IP地址。 最近上线时间：设备最近一次连接到系统的时间。 最新数据时间：设备最新数据上报的时间。 累计接收包总数：从设备接收到的总数据包数 当前缓存包数量：当前系统中缓存的设备数据包数量 设备发送包数：设备发送的总数据包数 累计接收字节数：设备接收到的数据字节总数 当前设备状态：显示设备当前的状态 累计上线次数：设备的累计连接次数 累计运行时长：设备累计在线时长 |
| IP设置 | √ |
| 设备详情 | X |
|  |  |
| 系统选项 | 自启动 |  |  |
| 数据存储路径（文件压缩、试验名文件夹） |  |  |
| 设置-允许触发定时同时 |  |  |
| 设置-全局采样 |  |  |
| 设置-包好对齐 | X | 这个F2才有的功能,不再实现 |
| 设置-输出到.DAT云振动格式 |  |  |
| 计算/转发 | 计算参数读取（从XML，兼容） |  |  |
| 物理量计算 |  |  |
| 索力计算 |  |  |
| 公式计算 |  |  |
| MQTT推送 |  |  |
|  |  |  |
| 故障/优化迭代 | 适配FS-V08: 频率去掉32K;去掉软触发字段判断 |  |  |
| 记录最近一次连接的时间；防止频繁的重连 |  |  |
| DAAS软件通信协议增加服务器主机时间下发 |  |  |
|  |  |  |
|  |  |  |

# 功能模块

## 设备列表

参考DAAS查看设备状态界面。支持在线设备的列表展示，并且在选中后，显示当前设备的通道配置信息。





## 采样控制

采样即达到某种触发条件后，将振动信号数据保存，持续保存到指定【采样点数】的数据点位后为一包（一个批次），供后续计算、展示使用。根据现场使用情况，应该支持以下几种方式的采样策略：

1. 通道触发

需要支持各个通道触发电平以及是否参与，支持触发判断方式（绝对值、上升沿、下降沿），支持滞后点数配置。



1. 定时触发

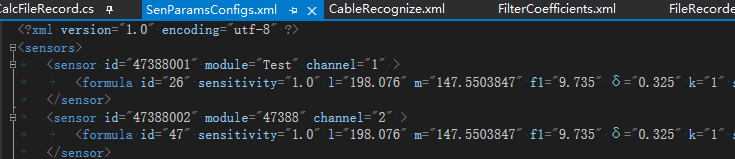
支持定时触发方式，设置定时器触发。可以配置采样次数，指定每次达到定时后，采集几个批次。

1. 连续触发

定时采集的一种特殊情况，连续不间断采集。

## 公式计算

支持从XML中读取传感器计算参数。如下是索力计算公式的参数。程序在初始化时候读取。



在EdgeDataHandler中注册处理方法，实现公式计算。

## 数据转发

支持默认MQTT方式上传数据。

统一上报topic主题格式 Upload/ZD/{SensorName}。

上报内容：

var jsondata = new

{

Sensorid = (int)sen.SensorID,

Module = \_ch.StationNo,//更新模块号为界面配置的测点号

Channel = sen.Channel,

Frequency= Convert.ToInt32(\_ch.SampleFreq),

ThemeValue = data.ThemeValues,

RawValues = new double[]{},

AccValues = data.VibData.Select(o=>Math.Round(o,3)).ToArray(),

Ticks = time //得到精确到秒的时间戳（长度10位）

};

以上SensorName和SensorId也是配置在SenParamsConfigs.xml文件中。

## 索力计算

索力计算目前使用的算法进行了调整，目前的索力值的计算，中间使用了较多的算法和逻辑（小波去噪、FFT、五点三次平滑、基频查找的详细逻辑、频谱倍增、去直流等）。修改后的索力计算流程如下：



修改后索力计算直接在频谱图中，在给定的频率范围内搜索最大值，再通过精确频率算法，得到N阶频率值，在除以N估算得到基频值，然后通过经典弦公式计算得到索力。