TOMFramework Project

Class Design Document

Version: 0.1 (Draft version)

Authors:

|  |  |  |
| --- | --- | --- |
| No. | Student ID | Student name |
| 33 | B2110131 | Nguyễn Lê Hoàng Lực |

SINH VIÊN ĐƯỢC PHÉP XEM TÀI LIỆU

SINH VIÊN KHÔNG ĐƯỢC SAO CHÉP HAY PHỔ BIẾN TÀI LIỆU TRONG PHÒNG THI DƯỚI MỌI HÌNH THỨC.

Contents

[1.](#_heading=h.30j0zll) Name: 4

[2.](#_heading=h.1fob9te) Problem 4

[3.](#_heading=h.3znysh7) Solution 4

[3.1.](#_heading=h.2et92p0) Class diagram 4

[3.2.](#_heading=h.tyjcwt) Class Descriptions 4

[3.2.1.](#_heading=h.3dy6vkm) Class: <Classname> 4

[4.](#_heading=h.1t3h5sf) Consequence 5

## Name: Sensor Status

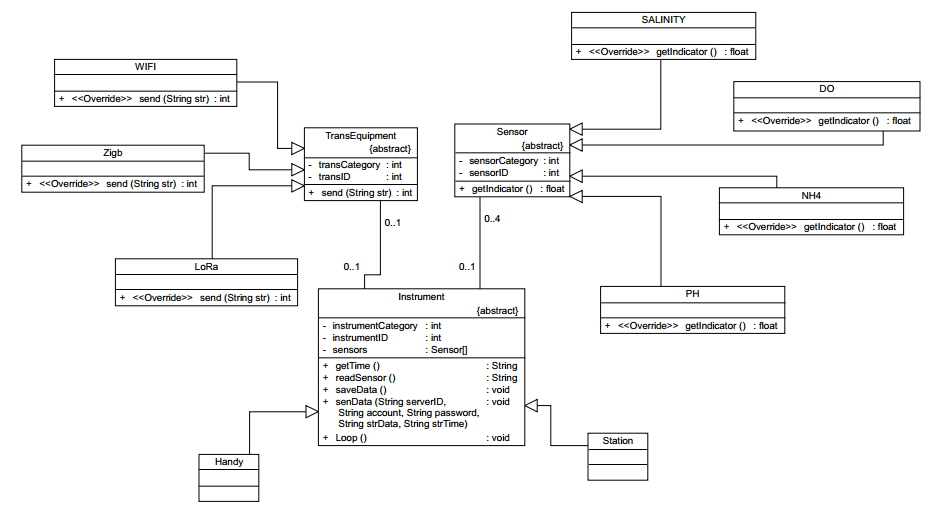
## Problem

Ngày nay việc quản lý thông số của các cảm biến trong hệ thống IoT đang ngày càng khó khăn khi ngày càng có nhiều loại cảm biến trong một hệ thống. Nhu cầu có một ứng dụng để tự động quản lý các thông số này đang cần thiết hơn bao giờ hết.

## Solution

Ứng dụng Sensor Status ra đời để giúp các doanh nghiệp IoT quản lý hiệu quả các thông số của các Sensor. Bằng cách tổng quát hóa các loại sensors, instruments, transEquipments thành những lớp trừu tượng. Từ đó ta có thể quản lý, phân loại cũng như truyền các thông số của sensor thông qua các thiết bị truyền khác nhau. Ngoài ra, ứng dụng giúp quản lý lưu trữ dữ liệu lên cơ sở dữ liệu một cách hiệu quả.

## Class diagram



## Class Descriptions

### Class: Instrument

* Purpose: *Lớp này đóng vai trò là lớp tổng quát cho các loại dụng cụ. Nó quy định nhũng thuộc tính và phương thức mặc định mà một công cụ cần có.*
* Constraints: *None*
* Persistent: *No (created at system initialization from other available data)*

Attribute Descriptions

1. *Attribute: instrumentCategogy*

Type: Integer

Description: *Mã loại dụng cụ*

Constraints: *None*

1. Attribute: instrumentID

Type: Integer

Description: *Mã dụng cụ*

Constraints: *None*

1. Attribute: sensors

Type: List<sensor>

Description: *Danh sách cảm biến được gắn vào công cụ*

Constraints: *None*

Method Descriptions

1. Method: getTime

Return Type: String

Parameters:None

Return value: *Chuỗi ký tự chứa thời gian*

Pre-condition: *Đọc giá trị thời gian thực*

Post-condition:

Đầu ra là chuỗi chứa thời gian theo định dạng YYYY-MM-DD-hh:mm-ss, trong đó

YYYY: bốn số chỉ năm MM: 02 số chỉ tháng DD: 02 số chỉ ngày

hh: 02 số chỉ giờ mm: 02 số chỉ phút ss: 02 số chỉ giây

Attributes read/used: *None*

Methods called: *None*

Processing logic:

Lấy thời gian thực từ hệ thống và trả về chuỗi chứa thời gian theo định dạng YYYY-MM-DD-hh:mm-ss

Test case:

*Kiểm tra hàm bằng cách in kết quả hàm ra console. Nếu* đầu ra là chuỗi chứa thời gian theo định dạng YYYY-MM-DD-hh:mm-ss thì hàm chạy đúng.

1. Method: readSensor

Return Type: String

Parameters:None

Return value: *Chuỗi ký tự chứa giá trị đọc từ các cảm biến*

Pre-condition: *Số lần gọi getIndicator() tuỳ thuộc vào số lượng cảm biến được gắn vào dụng cụ.*

Post-condition:

Giá trị trả về là chuỗi ký tự chứa giá trị đọc từ các cảm biến theo định dạng JSON.

Attributes read/used: *sensors*

Methods called: *getIndicator()*

Processing logic:

Hàm readSensor() lần lượt gọi hàm getIndicator() của mỗi cảm biến được gắn vào dụng cụ. Sau đó tiến hành đọc các giá trị này và trả về một chuỗi ký tự kết quả.

Test case:

*Kiểm tra hàm bằng cách in kết quả hàm ra console. Nếu* đầu ra chuỗi ký tự chứa giá trị đọc từ các cảm biến theo định dạng JSON thì hàm chạy đúng.

1. Method: saveData

Return Type: None

Parameters:data - chuỗi kết quả của hàm readSensor()

Return value: *None*

Pre-condition: None*.*

Post-condition: None.

Attributes read/used: *None*

Methods called:readSensor()

Processing logic:

Hàm lưu chuỗi kết quả của hàm readSensor() vào thẻ nhớ, .

Test case:

*Kiểm tra hàm bằng cách xem dữ liệu trong thẻ nhớ. Nếu dữ liệu hợp lệ thì hàm chạy đúng*

1. Method: sendData

Return Type: Integer

Parameters:

serverID: chuỗi xác định địa chỉ

account: chuỗi tài khoản kết nối

password: chuỗi mật khẩu

strData: chuỗi giá trị chuyền

strTime: chuỗi thời gian

Return value: Mã thông báo trạng thái truyền dữ liệu

Pre-condition: Hàm sendData() sẽ gọi hàm send() tương ứng của thiết bị truyền được gắn kết*.*

Post-condition:  *0 là truyền thành công, giá trị khác 0 là mã lỗi*.

Attributes read/used: *sensors*

Methods called:send()

Processing logic:

Hàm truyền dữ liệu cảm biến, truyền chuỗi kết quả của hàm readSensor() về Data Server. sendData() có 4 tham số đầu vào kiểu chuỗi (String). Giá trị trả về kiểu số nguyên (0: truyền thành công, giá trị khác 0 là mã lỗi). Hàm senData() sẽ gọi hàm send() tương ứng của thiết bị truyền được gắn kết.

Test case:

*Kiểm tra hàm bằng cách in kết quả hàm ra console. Nếu kết quả trả về 0 tức là hàm chạy đúng.*

1. Method: Loop

Return Type: None

Parameters:None

Return value: None

Pre-condition: None

Attributes read/used:

Methods called:getTime(), readSensor(), saveData(), sendData()

Processing logic:

Hàm cố định không đổi, lần lượt thực hiện các công việc: đọc thời gian, đọc

giá trị các cảm biến, lưu giá trị cảm biến lên thẻ nhớ, truyền dữ liệu cảm biến về

cho Data Server.

Test case:

*Kiểm tra hàm bằng cách chạy hàm. Nếu kết quả trên console trả về lần lượt các công việc đã định trước thì hàm chạy đúng.*

### Class: Handy

Được thừa kế từ lớp trừu tượng Instrument

### Class: Station

Được thừa kế từ lớp trừu tượng Instrument

## Consequence

## Implementation