|  |  |
| --- | --- |
| Project | **Specification of Sensor Interface for Cyber and Physical World**  <https://sagroups.ieee.org/2888/ > |
| Title | **Environmental sensor data formats** |
| DCN | **2888-20-0011-01-0001** |
| Date Submitted | **Feb. 23, 2020** |
| Source(s) | Sang-Kyun Kim, [goldmunt@gmail.com](mailto:goldmunt@gmail.com) (Myongji University)  Kyoungro Yoon, [yoonk@konkuk.ac.kr](mailto:yoonk@konkuk.ac.kr) (Konkuk University)  Sangkwon Jeong, [ceo@joyfun.kr](mailto:ceo@joyfun.kr) (Joyfun) |
| Re: |  |
| Abstract | This contribution illustrates the basic JSON schema structure for representing sensor information in the physical world in a standardized data format. Sensors in the physical world transform the detected data into a JSON instance that follows the JSON schema presented by this contribution and forwards it to the cyber world. Cyber world checks the adequacy of the sensor data by checking whether the sensor data in JSON format conforms to the standardized JSON schema. |
| Purpose | To start discussion on purpose of the standard |
| Notice | This document has been prepared to assist the IEEE 2888 Working Group. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein. |
| Release | The contributor grants a free, irrevocable license to the IEEE to incorporate material contained in this contribution, and any modifications thereof, in the creation of an IEEE Standards publication; to copyright in the IEEE’s name any IEEE Standards publication even though it may include portions of this contribution; and at the IEEE’s sole discretion to permit others to reproduce in whole or in part the resulting IEEE Standards publication. The contributor also acknowledges and accepts that IEEE 3079 may make this contribution public. |
| Patent Policy | The contributor is familiar with IEEE patent policy, as stated in [Section 6 of the IEEE-SA Standards Board bylaws](http://standards.ieee.org/guides/opman/sect6.html#6.3) <[http://standards.ieee.org/guides/bylaws/sect6-7.html#6](http://127.0.0.1:4664/cache?event_id=757737&schema_id=1&s=5X0vID10lu_E6yrIkWkNd4Wz2H8&q=hancock)> and in *Understanding Patent Issues During IEEE Standards Development* <http://standards.ieee.org/board/pat/faq.pdf> |

# Introduction

Environmental sensors measures intensity or quality of lights, noises, temperature, humidity, wind, gas, and dust. They are the basic sensors for acquiring the environmental information of the physical world spaces, and the data through them are important information for determining the environmental characteristics of cyber spaces. This contribution proposes a schema for defining sensor data input from environmental sensors.

# Schema for environmental sensor data

## Light sensor

### Syntax of light sensor data format

    "lightSensorType": {

      "type": "object",

      "properties": {

        "value": {

          "$ref:": "#/definitions/colorWType"

        },

        "unit": {

          "$ref": "#/definitions/unitType"

        },

        "color": {

          "%ref": "#/definitions/colorType"

        },

        "colorValue": {

          "%ref": "#/definitions/colorValueType"

        },

        "model": {

          "$ref": "#/definitions/colorSpaceType"

        }

      }

    },

    "colorWType": {

      "type": "array",

      "items": [

        {

          "type": "number"

        },

        {

          "type": "string",

          "pattern": "#[0-9A-Fa-f]{2}"

        }

      ]

    },

    "colorValueType": {

      "type": "array",

      "items": [

        {

          "type": "number",

        }

      ],

      "minItems": 3,

      "maxItems": 3

    },

    "colorSpaceType": {

      "type": "string",

      "enum": [

        "XYZ",

        "Yxy",

        "Lab",

        "Lch",

        "LUV",

        "HunterLab"

      ]

    },

firgure 1. Schema for light sensor

Figure 1 shows the data schema of lightSensor. Include *value*, *unit*, *color*, *colorValue*, and *model* as properties. *value* represents the detected light intensity. The *unit* is lux and is used when the light sensor indicates “RGBW (Red Green Blue White)”. The *value* can represent a color by referring to *colorWType*. For example, #F0 represents white. The *unit* is the unit of the detected light intensity. The *color* refers to the color of light detected by the light sensor. The *colorValue* represents the color of the light, depending on the type of *model*. For example, if *model* is XYZ (CIEXYZ), then it is written as X, Y, and Z in order. The *model* specifies the color space of the color sensor. The notation of *colorValue* changes depending on the model.

### Example of light sensor data instance

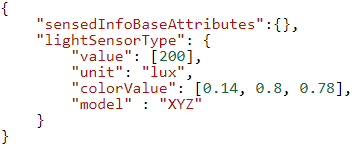


figure 2. Example of light sensor data JSON instance

Figure 2 shows the JSON instance of lightSensor. The light sensor detected 200 lux of light. Colors are represented in the CIE XYZ color space and represent blue with 0.14, 0.8, and 0.78.

## Ambient noise sensor

### Syntax of ambient noise sensor data format

    "ambientNoiseSensorType": {

      "type": "object",

      "properties": {

        "lifespan": {

          "type": "number"

        },

        "value": {

          "type": "number"

        },

        "unit": {

          "$ref": "#/definitions/unitType"

        }

      }

    }

figure 3. Schema for ambient noise sensor

Figure 3 shows the data schema of the ambient noise sensor. It includes *lifespan*, *value*, and *unit*. The *lifespan* represents the time taken to measure information based on the *timeStamp* of the root properties. The *lifespan* is the number of clocks in the sensor. The *value* is the measured ambient noise and the *unit* is the decibel (dB). The *unit* is the unit of ambient noise measured in dB if not specified.

### Example of ambient noise sensor data instance

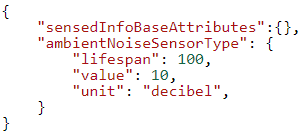


figure 4. Example of ambient noise sensor data JSON instance

Figure 4 shows the JSON instance of the ambientNoiseSensor. It is measured every 100 ticks of the clock in the ambient noise sensor. The ambient noise measured is 10 decibels.

## Temperature sensor

### Syntax of temperature sensor data format

    "temperatureSensorType": {

      "type": "object",

      "properties": {

        "value": {

          "type": "number"

        },

        "unit": {

          "$ref": "#/definitions/unitType"

        }

      }

    },

figure 5. Schema for temperature sensor

Figure 5 shows the data schema of the temperatureSensor. The *value* represents the temperature value and unit is C° (Celsius) if not specified.

### Example of temperature sensor data instance

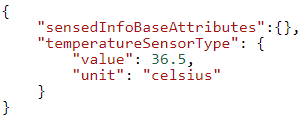


figure 6. Example of ambient noise sensor data JSON instance

Figure 6 shows the JSON instance of the temperatureSensor. In our example, the temperature sensor measured the temperature at 36.5 C˚.

## Humidity sensor

### Syntax of humidity sensor data format

    "humiditySensorType": {

      "type": "object",

      "properties": {

        "value": {

          "type": "number"

        },

        "unit": {

          "$ref": "#/definitions/unitType"

        }

      }

    },

figure 7. Schema for humidity sensor

Figure 7 shows the humiditySensor's data schema. The *value* measures humidity in percentage (%).

### Example of humidity sensor data instance

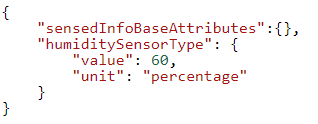


figure 8. Example of humidity sensor JSON instance

Figure 8 shows the JSON instance of the humiditySensor. The humidity sensor measured humidity at 60%.

## Wind sensor

### Syntax of wind sensor data format

    "windSensorType": {

      "$ref": "#/definitions/velocitySensorType"

    },

    "velocitySensorType": {

      "type": "object",

      "properties": {

        "velocity": {

          "$ref": "#/definitions/float3DVectorType"

        },

        "unit": {

          "$ref": "#/definitions/unitType"

        }

      }

    },

figure 9. Schema for wind sensor

Figure 9 shows the data schema of windSensor. The windSensorType refers to velocitySensorType. The velocitySensorType has *velocity* and *unit*. The *velocity* measures the velocity of a three-dimensional vector in meters per second (m/s).

### Example of wind sensor data instance

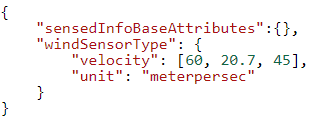


figure 10. Example of wind sensor JSON instance

Figure 10 shows the JSON instance of the windSensor. In the example, the wind velocity is measured at 60m/s in the x direction, 20.7m/s in the y direction, and 45m/s in the z direction.

## Gas sensor

### Syntax of gas sensor data format

    "gasSensorType": {

      "type": "object",

      "properties": {

        "gasType": {

          "type": "string",

          "enum": [

            "carbon monoxide",

            "carbon dioxide",

            "sulfurous acid",

            "nitrogen oxide",

            "oxygen",

            "ozone",

            "hydrogen",

            "VOC",

            "ethanol",

            "propane",

            "methane",

            "butane",

            "formaldehyde",

            "Radon222"

          ]

        },

        "value": {

          "type": "number"

        },

        "unit": {

          "$ref": "#/definitions/unitType"

        }

      }

    },

figure 11. Schema for gas sensor

Figure 11 shows the data schema of gasSensor. This includes *gasType*, *value*, and *unit*. The *gasType* selects one of the predefined gases. The *value* represents the gas concentration intensity. The *unit* uses parts per million (ppm) or picocuries per liter (pCipl).

### Example of gas sensor data instance

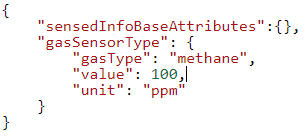


figure 12. Example of gas sensor JSON instance

Figure 12 shows the JSON instance of the gasSensor. The gas measured by the gas sensor is 100 ppm of methane gas.

## Dust sensor

### Syntax of dust sensor data format

    "dustSensorType": {

      "type": "object",

      "properties": {

        "value": {

          "type": "number"

        },

        "unit": {

          "$ref": "#/definitions/unitType"

        }

      }

    },

figure 13. Schema for dust sensor

Figure 13 shows the data schema of dustSensor. It contains value and unit. The unit defaults to *µ*g/m3 (microgram per cubic meters).

### Example of dust sensor data instance



figure 14. Example of dust sensor JSON instance

Figure 14 shows the JSON instance of dustSensor. The value measured by the dust sensor is 100 *µ*g/m3.

# Conclusion

It is recommended to adopt the JSON-based schema proposed in this contribution as the environmental sensor data schema of IEEE 2888.1. In addition, it is recommended to improve the data schema in the future by adding environmental sensor data information actually used in the industry.