INTEGRATION OF HETEROGENEOUS CORONAVIRUS DISEASE COVID-19 DATA SOURCES USING OGC SENSORTHINGS API

T. Santhanavanich*, C. Kim, V. Coors
University of Applied Sciences Stuttgart
Schellingstraße 24
70174 Stuttgart, Germany
(thunyathep.santhanavanich@hft-stuttgart.de, chaeyun.kim@outlook.com volker.coors@hft-stuttgart.de)

Commission IV

KEY WORDS: SensorThings, COVID-19, Smart Cities, Urban data, Open Geospatial Consortium

ABSTRACT:

The latest coronavirus (namely severe acute respiratory syndrome coronavirus 2 or COVID-19) was first detected in Wuhan, China, and spread throughout the world since December 2019. To tackle this pandemic, we need a tool to trace and predict trends of COVID-19 at global, national, and regional levels rapidly. Several organizations around the world offer access to COVID-19 related data. However, these data sources are heterogeneous in terms of data formats and protocols as different organizations developed them. To address this issue, a standard way to handle these datasets is needed. In this paper, we propose using the OGC SensorThings API to manage the COVID-19 dataset in a standard form and provide access to the general public. As a proof-of-concept, we implemented a COVID-19 data management platform based on the OGC SensorThings standard named COVID-19 SensorThings or in short COVID-STA. For a use case, we developed a real-time interactive web-based dashboard illustrating the COVID-19 dataset based on the COVID-STA. As a result, we proved that the OGC SensorThings API is suitable to use as a general standard for integrating the heterogeneous COVID-19 data.

1. INTRODUCTION

The novel coronavirus, namely severe acute respiratory syndrome coronavirus 2 or COVID-19, was detected in Wuhan, Hubei, China, in December 2019. It had spread rapidly, taking around one month to expand from Hubei to the rest of Main-land China (The Novel Coronavirus Pneumonia Emergency Re-sponse Epidemiology Team, 2020). Since then, the outbreak has spread throughout the world, causing more than 2.8 million confirmed cases worldwide as of 25 April 2020. In response to the fast-spreading of the fatal virus, many countries have ad-opted strict contact restriction measures such as lockdown and travel restriction (Ikejezie, 2020). The study suggested that it is essential to monitor the COVID-19 data frequently and accur-ately to spot the possible danger (Panelli, Piazza, 2020).

Accordingly, several academic organizations had developed GIS platforms for monitoring this pandemic. For example, the Johns Hopkins University’s Center for Systems Science and Engineer-ing (Dong et al., 2020) launched its dashboard (figure 1) since late January 2020. It is an interactive map that tracks confirmed infections, deaths, and recoveries all around the world, with the graphs showing the historical data of the spreading over time. However, this dashboard still misses the services to visualize map data from previous days, which is vital to see the spreading trend. Although the Johns Hopkins CSSE has provided its data-set for the general public, it is not the most updated resources as 1) the open data source only gets an update with limit times each day, and 2) there is some delay until the data in the local area is reported (Boulos, Geraghty, 2020). So, the platform should refer to the data from different sources to provide the dataset in different scales. This paper offers a method to handle this heterogeneity using the standard from OGC, SensorThings API, to integrate and share the COVID-19 data from several sources in a unified way.

The rest of this paper is organized as follows. Section 2 shows the background of SensorThings API. Section 3 describes the concept of our paper. Then, section 4 shows the implementa-tion of this research. Section 5 shows a use case as a proof-of-concept prototype. Then, section 6 discusses the result from several points of view. Finally, section 7 concludes the research outcome.

2. OGC SENSORTHINGS API

2.1 SensorThings API Standard

The SensorThings API is one of the OGC standards of a protocol that unifies ways to interconnect the Internet of Things

Figure 1. The global COVID-19 cases by Johns Hopkins CSSE, Screenshot by 25 April 2020
(IoT) devices, data, and applications over the Web (Liang et al., 2016). Several cities and departments had already provided their sensor data with this standard; for example, air quality data management in Europe by European environment agency, Smart City Sensors in Hamburg, water quality management of hydrological Network in Baden Württemberg and many more.

The SensorThings has two main parts which are Sensing and Tasking. In this paper, we focus only on the data management, which is the Sensing part, and to be extended with Tasking part in the future work. In the Sensing part, it provides an easy-to-use REST application programming interface (API). These operations are HTTP- POST, GET, PATCH, DELETE to create, read, update, and delete the sensor data and metadata, respectively.

### 2.2 SensorThings API Conceptual Model

The conceptual model of SensorThings API is based on the OGC Observations, and Measurement standard consists of eight entity types, as in figure [2] showing a simplified version of the SensorThings API entity model.

![Figure 2. A simplified SensorThings API entity model.](image)

In SensorThings API, a Thing is main entity that refers to any real-world physical object with its location maintained in the Location. When the Thing moves or relocates, its past location will be stored in HistoricalLocation. Each Thing has one or more Sensor that senses one or more data types. This data information is stored in ObservedProperty. Then, the Datastream is referred to a Thing with one particular Sensor and one particular ObservedProperty. It is used for storing a group of Observations which is a result from of the sensor that observes one particular FeatureOfInterest.

### 3. CONCEPT

This paper proposes a method to provide interoperability among several COVID-19 data providers and users globally to share the COVID-19 statistics in a unified way. It is time that the world should unify a way of managing open data. According to OGC (Reed et al., 2019), managing data in an interoperable way provides several benefits such as increasing data usability and efficiency and decreasing risks and costs for integrating data from various sources. Without the standard to keep the measurement data, each data provider uses different methods or standards to share their dataset, which each user or application has to implement their tools to use the dataset, as shown in figure [2].

By applying the standard for data management to the system architecture, observations, measurements, procedures, and metadata

1. https://airquality-frost.docker01.ilt-dmz.iosb.fraunhofer.de/v1.1
2. https://iot.hamburg.de/v1.0

For this reason, we propose using the SensorThings API as a standard to manage the COVID-19 dataset in a standard way. As a proof-of-concept, we implemented a SensorThings server that collected the COVID-19 dataset from different providers and developed the web application based on the SensorThings server. The details of the implementation are described in the next chapter.

### 4. IMPLEMENTATION

For the implementation of the COVID-19 SensorThings server (COVID-STA), figure [3] shows the overall system architecture. Further details of how this architect is implemented are explained as follows.

#### 4.1 Data Source

This section explains the COVID-19 data sources (figure [5]) used in the paper which are including 1) Johns Hopkins University Center for Systems Science and Engineering (JHU), 2) Worldometer, and 3) Robert Koch Institute (RKI), Germany. We plan to keep expanding the dataset over time. Although these three datasets are representing the COVID-19 data, their structures, coverage, formats, and level-of-details are different.

The details of each dataset are concluded in table 1.

#### 4.2 Implementing SensorThings API Server

In this research, we used the FROST-Server (Fraunhofer Open-source SensorThings Server), an open-source implementation

Table 1. List of the COVID-19 data sources

<table>
<thead>
<tr>
<th>Providers</th>
<th>Coverage</th>
<th>Level of details</th>
</tr>
</thead>
<tbody>
<tr>
<td>JHU</td>
<td>Globally</td>
<td>Country and State</td>
</tr>
<tr>
<td>Worldometer</td>
<td>Globally</td>
<td>Country</td>
</tr>
<tr>
<td>RKI</td>
<td>Germany</td>
<td>State and City</td>
</tr>
</tbody>
</table>

Table 2. List of the COVID-19 SensorThings entities.

<table>
<thead>
<tr>
<th>SensorThings Entity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thing</td>
<td>Administrative areas where the COVID-19 data is measured</td>
</tr>
<tr>
<td>Location</td>
<td>Central geo-coordinates of each administrative areas</td>
</tr>
<tr>
<td>Sensor</td>
<td>COVID-19 data providers (JHU, Worldometer, RKI, ...)</td>
</tr>
<tr>
<td>ObservedProperty</td>
<td>COVID-19 data types (Confirmed, Recovered, Deaths)</td>
</tr>
<tr>
<td>FeatureOfInterested</td>
<td>Administrative areas where the COVID-19 data is measured</td>
</tr>
<tr>
<td>Datastream</td>
<td>COVID-19 datastreams</td>
</tr>
<tr>
<td>Observation</td>
<td>COVID-19 time-series statistic data</td>
</tr>
</tbody>
</table>

4.3 SensorThings Data Modeling

In order to store the sensor data with the SensorThings, we had to model the data source to the SensorThings standard. The first four entities to be defined are Thing with its Location, Sensor, and ObservedProperty. The Thing refers to the administration area around the world, including 264 areas from JHU and 412 areas from RKI, and we take only total global updated cases from Worldometer. The Location of each Thing refers to its central geo-coordinates of that areas in geojson format. The Sensor refers to the data providers, including JHU, Worldometer, and RKI. The ObservedProperty refers to the different types of COVID-19 cases, including confirmed cases, recovered cases, and deaths. The registrations of these four entities are done with the POST request.

After the first four entities are registered to the COVID-STA server, we wrote the programming script to automatically register the Datastream which match the particular Thing, Sensor, and ObservedProperty together. Then, the COVID-STA is ready for receiving a measured data through the_observation entity without the need to specify a FeatureOfInterested because it refers to the same area of the Location in the COVID-19 use case. Overall, table 2 concludes list of all SensorThings entities with their descriptions. Examples of these entities in JSON format are shown in listing 2-7 in appendix A.

4.4 SensorThings Data Manager Tool

As the original dataset are heterogeneous, we created a tool to import and clean and transform all incoming dataset to match the SensorThings entity in JSON format and then store in the SensorThings server. This tool is called SensorThings Data
Manager Tool or in short STA Manager (figure 5-4). It is written on the Node.js and running on the server. It periodically fetches all datasets every one hour with the support of incoming .csv or .txt datasets. If the tool detects any update, then it continues with data cleaning, transforming, and storing the data to the COVID-STA server (figure 5-5).

5. USE CASE

As proof of concept, we developed an interactive web-based application named COVID-19-Dashboard. It aims to be a preparation measure of rapidly changing virus cases, raising awareness at the regional and national levels. It visualizes and tracks reported COVID-19 cases, including confirmed cases, recovered cases, deaths, and active cases, which are updated hourly using the data from the implemented COVID-STA server. Example user interfaces on mobile and PC are shown in figure 6.

![Figure 6. The COVID-19-Dashboard web application.](image)

This application is also implemented to demonstrate the benefits of using SensorThings for reporting the COVID-19 statistics data in multiple levels from a single endpoint in a standard way that was integrated from several sources. For example, the chart in figure 7 demonstrates the updated COVID-19 statistics data in the country level, and figure 8 shows the graph of COVID-19 statistics in the state level in Germany.

![Figure 7. An example chart showing the confirmed and recovered COVID-19 cases in country level; ordered by the number of confirmed cases (from COVID-19-Dashboard web application).](image)

![Figure 8. A chart showing daily new confirmed COVID-19 cases in the state level of Germany from the COVID-19-Dashboard web application.](image)

As data from the COVID-STA server contains the geospatial location of all Datastreams according to the SensorThings standard, the statistic data can be represented spatially in the 2D or 3D maps. Especially in the modern web-based application, the data can be visualized in the dimension of time-space and location-space using map animation. For example, figure 9 illustrates the COVID-19 statistic data in Germany as the animated 3D map on the Kepler.gl web application.

![Figure 9. An example animated 3D map illustrating the timely confirmed COVID-19 cases in Germany using the data from COVID-STA service; visualized with the Kepler.gl application.](image)

6. DISCUSSION

The OGC SensorThings API standard is implemented to manage the COVID-19 dataset from different data sources. This server is called COVID-STA. It shares COVID-19 statistical data from multiple levels of administrative area. The researchers and developers can use the data from the COVID-STA service to develop a platform or application to distribute the COVID-19 statistics. They can also continue research about the COVID-19 and other factors that might affect the spreading speed, such as air pollution, geospatial location, weather condition, etc.

For the perspective of the COVID-STA implementation, the FROST SensorThings implementation provided by Fraunhofer helps us deploy the SensorThings application without any issue. We also used the extended SensorThings v1.1 which provides the property fields to the Sensor and Datastream entities which are necessary to give the metadata such as data provider information in Sensor or data accuracy in Datastream. However, some important features are still missing, such as 1) a feature to request data in another common standard such as a

---

5. https://covid19dashboard.org/

6. https://kepler.gl/

cav file format, 2) a feature to request aggregated data from SensorThings which significantly helps to reduce bandwidth between servers and users in large-scale. In future work, we will also extend the current implementation with the Tasking part of SensorThings.

For the data collection process, we successfully implemented a tool to integrate dataset fetching from data sources in hourly intervals and post to the SensorThings server. Even though the more frequent fetching process can be performed; however, the ideal solution in the future would be having access to the SensorThings interfaces directly from sources. Concerning the geospatial data, we stored the central coordinates of all administrative areas in our COVID-STA in the Locations entity as a point. The polygon data representing the administrative area can be added to Locations entity to represent the administrative area of each area more precisely.

For the user perspective, users can request the COVID-19 dataset and its metadata from the SensorThings server with HTTP GET request. The data query in SQL through this request is possible with the filter keys, which gives the flexibility to request the specific dataset in a given period. Additionally, users can access other datasets hosted with the SensorThings interface in the same way and integrate it into their application. Several open datasets in Europe already had their data this way, as mentioned in chapter 2. Importantly, more public sectors should consider sharing the COVID statistics or other open data in a unified approach, which we proved in this paper that the SensorThings could be used as a standard protocol for this requirement.

For the use case, we developed a real-time interactive web-based dashboard to track COVID-19 named COVID-19-Dashboard which fetches the data from the COVID-STA server as a single endpoint. This tool can monitor the trends in COVID-19 at the global, national, and regional levels and rapidly detect new cases so that the response section can pay close attention to the area to conduct risk management and help prepare countermeasures.

7. CONCLUSION

Overall, our study shows the use of the OGC SensorThings API standard to integrate and distribute the geostatistic data of COVID-19 cases as a time-series with a single endpoint. We successfully implemented the COVID-19 SensorThings server (COVID-STA) using FROST-server and integrated the COVID-19 geostatistic data from several sources in multiple administrative levels. It is essential to use reliable sources of data and always updates the metadata in the SensorThings entity. In the next step, it is relevant to promote the use of SensorThings so that the public sectors consider using the SensorThings as their standard protocols, which will benefit the researchers and developers in using the open sensor or statistic data in a unified way.

ACKNOWLEDGEMENTS

This work has been jointly developed in the project i_city (funding number: 03FH91011A). The project i_city is supported by the German Federal Ministry of Education and Research (BMBF). The authors are responsible for the content of this publication. The authors would like to thank Kathi Schleidt and Hylke van der Schaaf for their valuable suggestions and comments to improve the quality of COVID-STA server. Also, we would like to thank the anonymous reviewers who have helped to increase the quality of the paper.

REFERENCES


Reed, C., Botts, M., Percivall, G., Davidson, J., 2019. OGC Sensor Web Enablement: Overview And High Level Architecture. docs.opengeospatial.org/wp/07-165r1/


APPENDIX

Appendix A: Examples of the SensorThings Entities in COVID-STA server

Listing 2. An example of Thing entity in COVID-19 SensorThings server.

```json
"name": "Australian Capital Territory - Australia",
"description": "A Thing representing Australian Capital Territory - Australia area",
"properties": {
```

http://covid19dashboard.org
"country": "Australia",
"ISO_countryCode_2": "AU",
"ISO_countryCode_3": "AUS",
"ISO_Numeric_code": "036",
"province": "Australian Capital Territory"
}

Listing 3. An example of Location entity in COVID-19 SensorThings server.

{
"name": "Australian Capital Territory-Australia",
"description": "The central coordinates of the Australian Capital Territory, Australia in EPSG-4326",
"encodingType": "application/vnd.geo+json",
"location": {
"type": "Point",
"coordinates": [-35.4735, 149.0124]
}
}


{
"name": "COVID-19 dataset from Johns Hopkins CSSE",
"description": "Synced COVID-19 dataset from Johns Hopkins CSSE GISandData Github Repository",
"properties": {
"url": "https://github.com/CSSEGISandData/COVID-19",
"organisationName": "Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE)",
"Contact": "jhusystems@gmail.com"
},
"encodingType": "application/pdf",
"metadata": "https://www.thelancet.com/action/showPdf?pii=S1473-3099%2820%2930120-1"
}


{
"name": "Confirmed COVID-19 Cases",
"description": "Total Confirmed COVID-19 Cases",
"definition": "https://systems.jhu.edu/research/public-health/ncov/"
}


{
"name": "Confirmed COVID-19 Cases",
"description": "A dataset of confirmed COVID-19 Cases in Australian Capital Territory",
"observationType": "http://www.opengis.net/def/observationType/OGC-OM/2.0/OM_Measurement",
"phenomenonTime": "2020-04-01T00:00:00.000Z/2020-04-16T00:00:00.000Z",
"resultTime": "2020-04-01T00:00:00.000Z/2020-04-16T00:00:00.000Z",
"unitOfMeasurement": {
"name": "Number of Cases",
"symbol": "-",
"definition": "https://en.wikipedia.org/wiki/Unit_of_measurement"
}
}


{
"phenomenonTime": "2020-04-14T00:00:00.000Z",
"resultTime": "2020-04-14T00:00:00.000Z",
"result": 2613,
"@iot.id": 13720
}
{
"phenomenonTime": "2020-04-13T00:00:00.000Z",
"resultTime": "2020-04-13T00:00:00.000Z",
"result": 2579,
"@iot.id": 13719
}

Appendix B: Examples of the HTTP GET request to receive data from COVID-STA server.

Listing 8. An example HTTP GET request to receive datastreams with only observations in Germany (filtering datastreams where the Things’s name equal to ‘Germany’).

http://[IP Address]:[Port]/[STA Name]/[Version]/Datastreams
?$filter=Things/name eq 'Germany' &$expand=Observations

Listing 9. An example HTTP GET request to receive datastreams with only observations from JHU CSSE (filtering datastreams where the Sensor’s name contains ‘Johns Hopkins’).

http://[IP Address]:[Port]/[STA Name]/[Version]/Datastreams
?$filter=substringof('Johns Hopkins',Sensor/name) &$expand=Observations
Listing 10. An example HTTP GET request to receive datastreams for the number of confirmed COVID-19 cases from Robert Koch Institut (filtering datastreams where the Sensor’s name contains ‘RKI’ and the ObservedProperties’s name contains ‘Confirm’).

```
http://[IP Address]:[Port]/[STA Name]/[Version]/Datastreams
?filter=substringof('RKI',Sensor/name) and substringof('Confirm',ObservedProperties/name)
&expand=Observations
```