CEN WORKSHOP AGREEMENT

CWA 18096

April 2024

ICS 07.140; 35.110

English version

CBRNe SENSOR API - Network Protocols, Data Formats and Interfaces

This CEN Workshop Agreement has been drafted and approved by a Workshop of representatives of interested parties, the constitution of which is indicated in the foreword of this Workshop Agreement.

The formal process followed by the Workshop in the development of this Workshop Agreement has been endorsed by the National Members of CEN but neither the National Members of CEN nor the CEN-CENELEC Management Centre can be held accountable for the technical content of this CEN Workshop Agreement or possible conflicts with standards or legislation.

This CEN Workshop Agreement can in no way be held as being an official standard developed by CEN and its Members.

This CEN Workshop Agreement is publicly available as a reference document from the CEN Members National Standard Bodies.

CEN members are the national standards bodies of Austria, Belgium, Bulgaria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Türkiye and United Kingdom.

EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

CEN-CENELEC Management Centre: Rue de la Science 23, B-1040 Brussels

Contents

Page

European foreword

This CEN Workshop Agreement has been developed in accordance with the CEN-CENELEC Guide 29 "CEN/CENELEC Workshop Agreements – A rapid prototyping to standardization" and with the relevant provisions of CEN/CENELEC Internal Regulations – Part 2. It was approved by a Workshop of representatives of interested parties on 2022-10-11, the constitution of which was supported by CEN following the public call for participation made on 2022-09-08. However, this CEN Workshop Agreement does not necessarily include all relevant stakeholders.

This workshop was coordinated by Yusuf Yilmaz and Ulrike Schröder (DIN), chaired by Marco Manso (PARTICLE) and co-chaired by Roberto Chirico (ENEA).

The final text of this CEN Workshop Agreement was provided to CEN for publication on 2024-04-10.

Results incorporated in this CWA received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 883116 (project RISEN - Real-time on-site forensic trace qualification).

The following organizations and individuals developed and approved this CEN Workshop Agreement:

— PARTICLE SUMMARY Lda. (Marco Manso)

— Agenzia nazionale per le nuove tecnologie, l'energia e lo sviluppo economico sostenibile (Roberto Chirico)

— Consorzio Creo-Centro Ricerche Elettro Ottiche (Nicola Liberatore)

— MASA Tech S.R.O. (Martin Sabo)

— Raggruppamento Carabinieri Investigazioni Scientifiche (Fabio Gianni Voria, Giuliano Iacobellis)

— Studio Indagini Mediche E Forensi (Anna Barbaro, Angelo La Marca)

— Universidad de Alcala (Carmen García-Ruiz, Cristina Cano Trujillo, Gemma Montalvo García)

— Wojskowa Akademia Techniczna Im.Jaroslawa Dabrowskiego (Bartłomiej Jankiewicz, Maksymilian Wlodarski, Miron Kaliszewski, Jaroslaw Mlynczak)

Secretariat: DIN e. V. (Yusuf Yilmaz, Ulrike Schröder)

Attention is drawn to the possibility that some elements of this document may be subject to patent rights. CEN-CENELEC policy on patent rights is described in CEN-CENELEC Guide 8 "Guidelines for Implementation of the Common IPR Policy on Patent". CEN shall not be held responsible for identifying any or all such patent rights.

Although the Workshop parties have made every effort to ensure the reliability and accuracy of technical and non-technical descriptions, the Workshop is not able to guarantee, explicitly or implicitly, the correctness of this document. Anyone who applies this CEN Workshop Agreement shall be aware that neither the Workshop, nor CEN, can be held liable for damages or losses of any kind whatsoever. The use of this CEN Workshop Agreement does not relieve users of their responsibility for their own actions, and they apply this document at their own risk. The CEN Workshop Agreement should not be construed as legal advice authoritatively endorsed by CEN/CENELEC.

Introduction

Traditional forensic examinations are usually time consuming which can be problematic when investigations are underway and quick results are needed. In the case of laboratory analysis of traces, these can take hours or days and may result in the destruction of the sample. It is important to detect and analyse traces on site as soon as possible before they degrade and loose important forensic information.

The Real-tIme on-site forenSic tracE qualificatioN (RISEN) action is funded by the European Commission (call H2020-SU-SEC-2019, topic SU-FCT02-2019, grant agreement number 883116) aiming to develop several real-time mobile analytical instruments (herein also referred as sensors) for the optimisation of trace detection, visualisation, identification and interpretation on site. RISEN processes data in real time, producing a 3D augmented crime scene investigation system with an interactive 3D model of the scene with position, labelling of traces and relative analytical results. RISEN operates as a network-enabled system where its constituting components are connected and exchange data pertaining to the forensic examination performed on site. Being a novel field, the lack of available standards results in manufacturers following their own implementation path, which in turn results in non-interoperable sensors that negatively impact on the efficiency of what is by nature an already time-consuming activity.

As part of the RISEN Action and with the support of the German Institute for Standardization (DIN), it was launched the CEN Workshop Agreement to produce the document "CBRNe SENSOR API — Network protocols, data formats and interfaces" (hereby called "RISEN API") enabling different forensics sensors to communicate in a consistent and harmonised way with a remote server, using widely used web-based technologies. This document defines the application programming interface (API) as a mechanism that allows sensors, as well as software components in general, to operate with a compliant remote server in a modular way. This means that, by following the RISEN API specifications, any authorised component can seamlessly connect to and exchange information with the RISEN System.

The RISEN API was implemented and demonstrated as part of the RISEN Action involving the several sensors developed in RISEN.

This proposed document has relation with the following:

- ISO 21043-1:2018, *Forensic sciences — Part 1: Terms and definitions*
- ISO 21043-2:2018, *Forensic sciences — Part 2: Recognition, recording, collecting, transport and storage of items*
- ISO/CD 21043-3, *Forensic Sciences — Part 3: Analysis*[1]
- ISO/CD 21043-4, *Forensic Sciences — Part 4: Interpretation*[1]
- ISO/CD 21043-5, *Forensic Sciences — Part 5: Reporting*[1]
- ISO/IEC 30128:2014, *Information technology — Sensor networks — Generic Sensor Network Application Interface*
- ISO/IEC series 29182, *Information technology — Sensor networks: Sensor Network Reference Architecture (SNRA).*

This document results from work conducted as part for RISEN Action and scientific publications as follows:

- Manso, Marco; Chirico, Roberto; Peltola, Johannes; Engström, Philip; Larsson, Håkan; Berggren, Jimmy. (2020) The RISEN Project – A Novel Concept for Real-time on-site Forensic Trace

[1] Currently under development within ISO/TC 272 Forensic sciences

Qualification. International Command and Control Research and Technology Symposium (ICCRTS). 25th ICCRTS Proceedings. https://doi.org/10.5281/zenodo.4264926

— Manso, M.; Guerra, B.; Freire, F.; Chirico, R.; Liberatore, N.; Linder, R.; Schröder, U. and Yilmaz, Y. (2023). On the Path Towards Standardisation of a Sensor API for Forensics Investigations. In Proceedings of the 12th International Conference on Sensor Networks - SENSORNETS, ISBN 978-989-758-635-4; ISSN 2184-4380, pages 15-22. DOI: 10.5220/0011688300003399

— RISEN Consortium. D4.3 RISEN Sensor API Definition Document v1.0. Dated: 30-08-2021

1 Scope

This document describes a Sensor Application Programming Interface (API) enabling a set of network-enabled near real-time contactless sensors, used in the context of crime scene investigations, to connect to the RISEN System for the optimisation of the trace, detection, visualisation, identification and interpretation on-site, combining 3D scene reconstruction capabilities and digital evidence management.

It defines the application programming interface (API) for the RISEN System as the mechanism to allow the RISEN sensors (or analytical tools) to operate with the RISEN System in a modular way. This means that, by following the RISEN API specifications, and subject to successful authorisation, any sensor can seamlessly connect to and exchange information with the RISEN System.

The API concept is designed to offer flexibility, modularity and interoperability by incorporating widely used Internet-based standards and technologies. Moreover, the definition of a “Generic API”, incorporating a “common interface bus” and common sensor functions, allows for different RISEN sensors to interface with the RISEN System in a harmonised way, thus enabling the “RISEN Sensor API”-compliant sensors to seamlessly connect to and exchange information (i.e., plug’n’play) with the RISEN System.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org

3.1
API
Application Programming Interface
type of software interface, offering a service to other pieces of software, acting as a connection between computers or between computer programs

3.2
API specification
document or standard that describes how to build an API connection or interface

3.3
chain of custody
chronological record of the handling and storage of an item from its point of collection to its final return or disposal

Note entry:Chain of custody is one element that contributes to the integrity of an item.

[SOURCE: ISO 21043-1:2018, 3.1]

3.4
classification
attribution of an object or trace to a definite class

[SOURCE: [1]]

3.5
collection
process of detecting, documenting, or retaining physical evidence

[SOURCE: [3]]

3.6
crime scene
scene of incident prior to establishing whether a criminal or illegal action has taken place or not

Note 1 to entry: The crime scene is not solely restricted to the location of the incident, but also includes areas where relevant acts were carried out before or after the crime. Suspects and victims who are subject to an examination for the recovery of forensic and/or medical evidence can also be considered to be crime scenes.

[SOURCE: [2]]

3.7
database
digitally available collection of data but also reference data

EXAMPLE Maps, citable literature and related information, sample collection, geographic information system (GIS) databases, identification keys, etc.

[SOURCE: [4]]

3.8
detection
<drug analysis> recognition of a drug or class of drugs using an analytical technique

[SOURCE: [5]]

3.9
digitalised evidence
digitized evidence
digitization of physical traces from crime scenes in forensic investigations to create a digital chain of custody

Note 1 to entry: The digitization of physical traces from crime scenes in forensic investigations in effect creates a digital chain-of-custody and entrains the challenge of creating a link between the two or more representations of the same trace. In order to be forensically sound, especially the two security aspects of integrity and authenticity need to be maintained at all times. Especially the adherence to the authenticity using technical means proves to be a challenge at the boundary between the physical object and its digital representations.

[SOURCE: [6]]

3.10
evidence
material items or assertions of fact that may be submitted to a competent tribunal as a means of ascertaining the truth of any alleged matter of fact under investigation before it

[SOURCE: [7]]

3.11
examination
act or process of observing, searching, detecting, recording, prioritizing, collecting, analysing, measuring, comparing and/or interpreting

Note 1 to entry: Examination can include collecting items from persons.

[SOURCE: ISO 21043-1:2018, 3.9]

3.12
identification
first two steps of the process of interpreting the scientific evidence or trace: (1) classification and (2) individualisation.

[SOURCE: [1]]

3.13
individualisation
possibility to distinguish one object or trace from all objects or traces considered

[SOURCE: [1]]

3.14
measurement rate
reported points per second

[SOURCE: [8]]

3.15
measurement scale
object showing standard units of length (e.g., ruler) used in photographic documentation of an item of evidence

[SOURCE: [3]]

3.16
OpenAPI Specification
OAS
standard and language-agnostic interface to RESTful APIs, allowing software clients to understand the capabilities of the service, while at the same time presenting it to humans in a user-friendly way

3.17
sample
portion drawn from a whole or population for the purpose of examination/testing, not necessarily representative of the whole

[SOURCE: ISO 21043-1:2018, 3.28, modified – Note 1 to entry and Example have been deleted.]

3.18
trace
most elementary information that results from crime

[SOURCE: [9]]

4 Abbreviated terms

3DA-CSI	3 Dimensional Augmented Crime Scene Investigation
API	Application Programming Interface
COTS	Commercial Off The Shelf
CSI	Crime Scene Investigation
HTTP(S)	Hypertext Transfer Protocol (Secure)
IP	Internet Protocol
ISP	Internet Service Provider
JSON	JavaScript Object Notation
LEA	Law Enforcement Agency
MQTT	Message Queuing Telemetry Transport
OAS	OpenAPI Specification
REST	Representational State Transfer
RFC	Request For Comments
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UI	User Interface
URI	Uniform Resource Identifier

UTC	Coordinated Universal Time
UTP	Unshield Twisted Pair
VPN	Virtual Private Network
VR	Virtual Reality
WE	Work Element
Wi-Fi	Wireless Fidelity
XML	eXtensible Markup Language

5 RISEN Sensor API

5.1 General

The Sensor API enables a set of network-enabled near real-time contactless sensors to connect to the RISEN System and send trace and measurement data collected from a crime scene. The Sensor API comprises a server-side component, which is receives and manages data generated by sensors, and a client-side component, which is placed at the sensor's side and sends sensor data to the server component.

In order to obtain a high degree of interoperability and compatibility, the server-client connection supports widely adopted and available Internet-based technologies

5.2 RISEN Sensor API Architecture

The sensor API concept includes a server-side component (i.e., the 3DA-CSI) and a client-side component (i.e., RISEN API Client). The client-side component should comply with the RISEN Sensor API specifications in order to be able to send sensor data. Sensor specific functions can be implemented by internally extending and implementing the necessary logic, while keeping the interface specifications aligned with the RISEN Sensor API.

Figure 1 shows the RISEN Sensor API high level architecture that is described next.

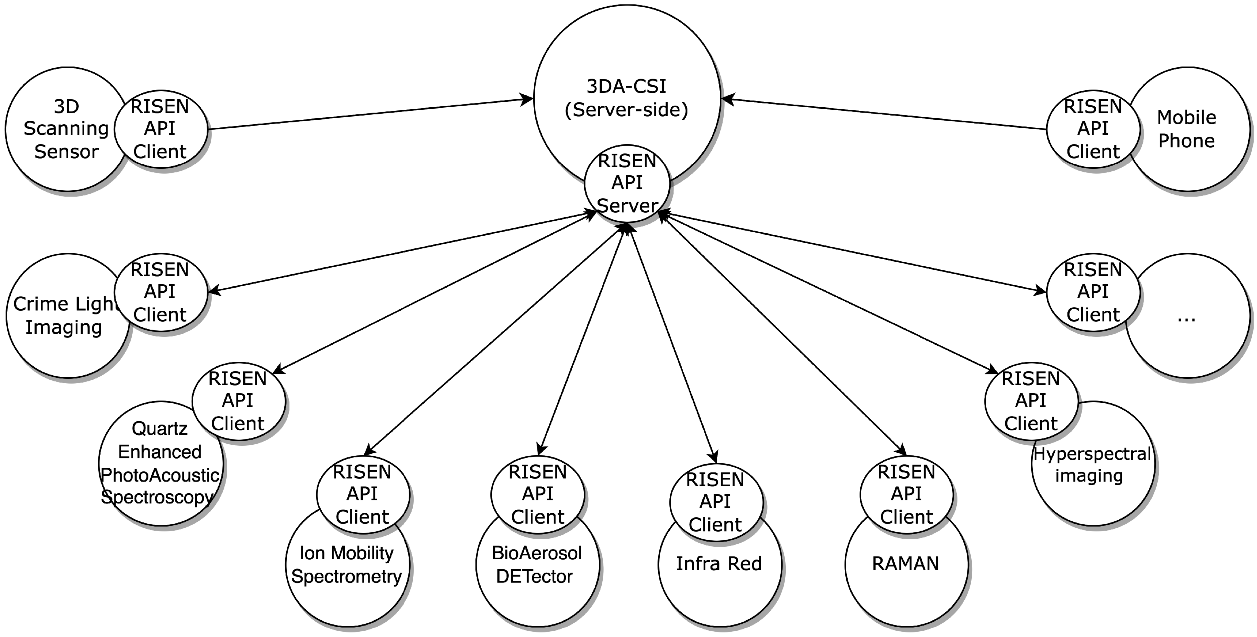


Figure 1 — API Architecture

The RISEN API follows a server-client architecture.

On the server side and placed at the center of Figure 1, there is the RISEN API Server that resides in the RISEN System server. The API Server is accessible by clients using the server's Uniform Resource Identifier (URI).

On the client side, several classes of sensors implement and extend the RISEN Sensor API (as in the case for sensors manufactured by RISEN partners) or, in case of commercial-off-the-shelf (COTS) products (e.g., 3D scanning sensors, single-lens reflex cameras and mobile phones), can rely on 3rd party software that implements the API and thus integrate their information in the RISEN System.

The server-client connection supports Internet-based technologies for maximum interoperability and compatibility, including the use of the Internet Protocol (IP) and the Transmission Control Protocol (TCP)/User Datagram Protocol (UDP) for data exchange, and will be realised via wired (e.g., Unshield Twisted Pair (UTP) or optical cable) or wireless (e.g., IEEE802.11ac/Wireless Fidelity (Wi-Fi)) connectivity.

5.3 RISEN Sensor API Implementation Approach

5.3.1 General

The approach for the RISEN Sensor API is based on open-source and widely used technologies, as presented next.

5.3.2 HTTP, JSON and REST

The RISEN Sensor API follows the REpresentational State Transfer (REST) architecture. The client-side component can send requests to the server-side component through the server URI, using the Hypertext Transfer Protocol (Secure) (HTTP(S)) protocol, including payload data typically formatted as JavaScript Object Notation (JSON), eXtensible Markup Language (XML) or plain text. The RISEN System will favour the use of JSON for payload data.

The HTTP protocol will be used including the methods described below [11]:

- GET can be used to request transfer of a current selected representation for the target resource. It is the primary mechanism of information retrieval.
- POST requests that the target resource processes the representation enclosed in the request. It is typically used as a mechanism to upload information.
- PUT requests that the state of the target resource be created or replaced with the state defined by the representation enclosed in the request message payload.
- DELETE requests that the origin server removes the association between the target resource and its current functionality.

HTTP request produces a status code indicating if the request was successful or if an error occurred. The status codes are categorised as follows:

- 1xx (Informational): The request was received, continuing process.
- 2xx (Successful): The request was successfully received, understood, and accepted.
- 3xx (Redirection): Further action needs to be taken in order to complete the request (i.e., the operation should be retried).
- 4xx (Client Error): The request contains bad syntax or cannot be fulfilled (e.g., bad request or unauthorised request).

— 5xx (Server Error): The server failed to fulfil an apparently valid request.

Future versions of the API will consider additional technologies, like the use of message brokers based on the publish-subscribe paradigm.

5.4 Deployment Options

5.4.1 General

The network-enabled architecture selected for the RISEN System and the Sensor API, supported by widely adopted and open web-standards, delivers a high degree of flexibility concerning the deployment options for the RISEN System in operational environments.

This subsection presents the deployment options for the RISEN System, allowing to meet most LEA scenarios.

5.4.2 RISEN deployment at the crime scene

In this setting, the RISEN System is transported and installed at the crime scene. This includes all RISEN sensors and the 3DA-CSI System component.

A local network (wired or wireless) is established to connect all RISEN components. The RISEN sensors send measurements to the 3DA-CSI System component. The investigator accesses the 3DA-CSI System locally, using a computer or an App. All information is locally stored in the 3DA-CSI System component.

In this setting, it is not possible to remotely access information from the 3DA-CSI System (e.g., request assessment from an expert not located at the crime site).

Users can export all gathered data to a central 3DA-CSI System a posteriori, using the RISEN 3DA-CSI System’s export/import functions.

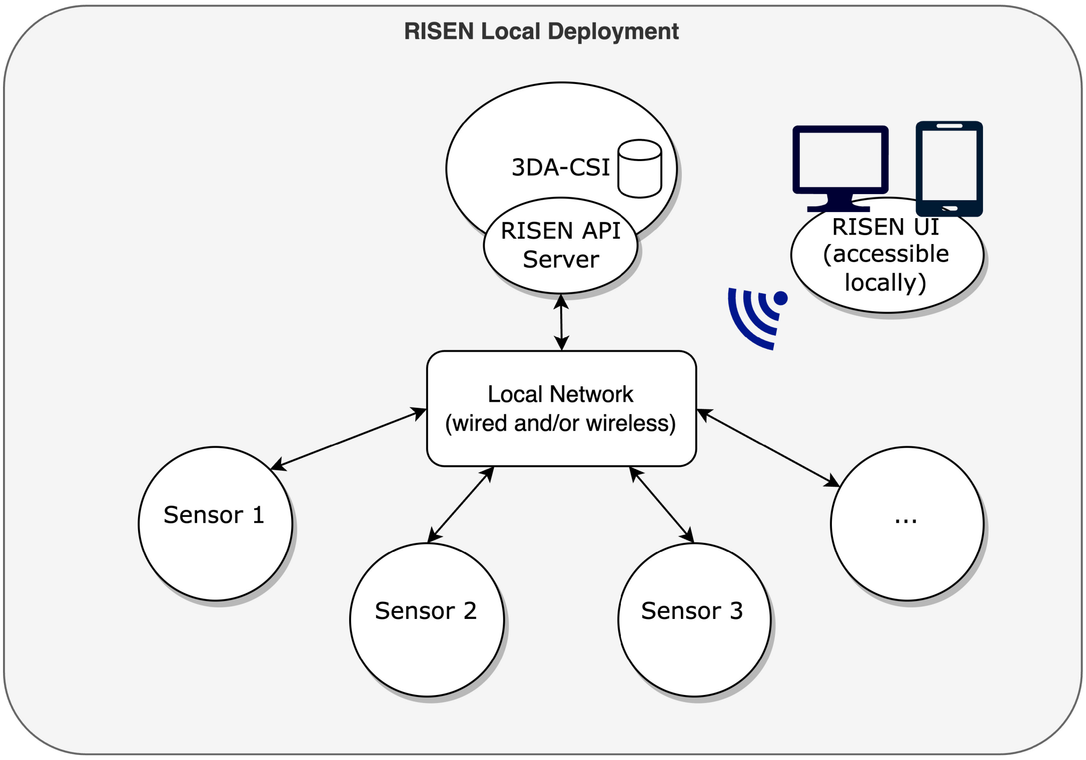


Figure 2 — Local Deployment

This setting is advantageous for situations where network connectivity with a central 3DA-CSI System is not possible (or desired). However, the setting also limits the availability of gathered data in near-real time to investigators on-site only. Moreover, the 3DA-CSI System component should be running on a portable device, with limited processing capabilities.

5.4.3 RISEN deployed in a vehicle near the crime scene

In this setting, the RISEN System is transported and installed at the crime scene. However, while the RISEN sensors are placed at the crime scene, the 3DA-CSI System component remains in a vehicle, stationed near the crime site, thus delivering good processing and power capabilities.

A local wireless network is established to connect all RISEN components. Where needed, network repeaters are installed to ensure adequate network coverage.

The RISEN sensors send measurements to the 3DA-CSI System component. The investigator accesses the 3DA-CSI System at the crime site using a computer in the vehicle or, via a portable computer or App, at the crime scene. All information is locally stored in the 3DA-CSI System component.

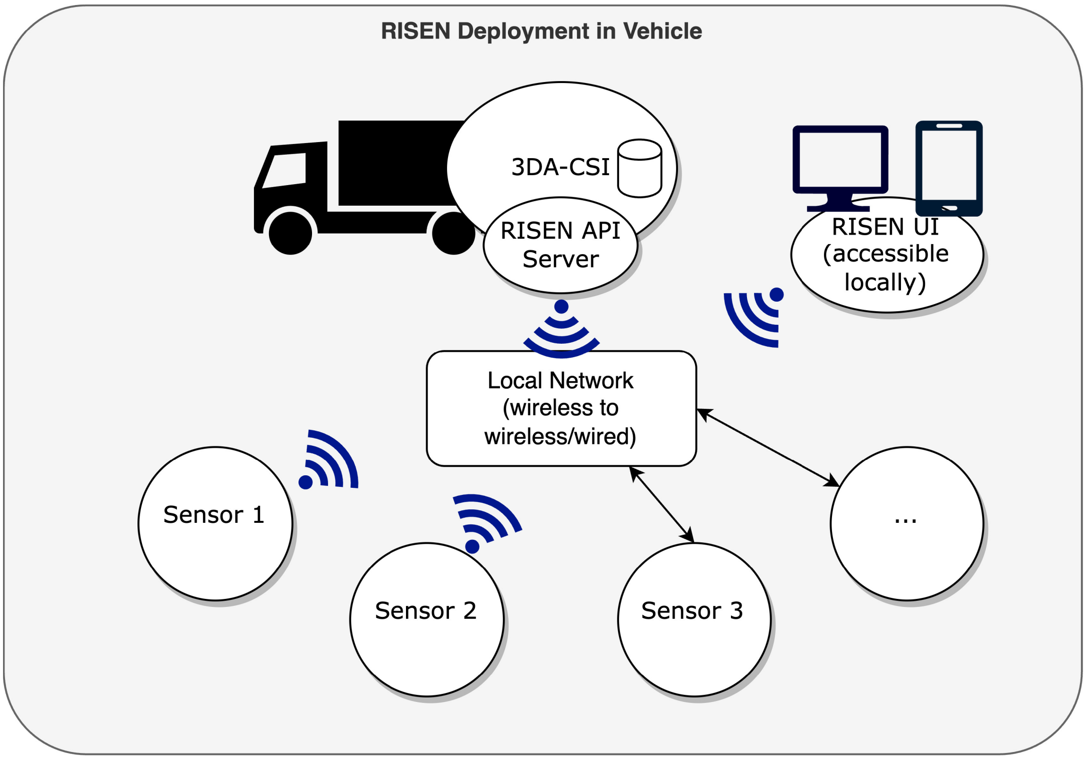

RISEN Deployment in Vehicle
3DA-CSI
RISEN API Server
RISEN UI (accessible locally)
Local Network (wireless to wireless/wired)
Sensor 1
Sensor 2
Sensor 3
...

Figure 3 — Deployment in Vehicle

Compared with the setting “RISEN deployment at the crime scene”, this one brings as main advantage the increased computing and power capabilities provided by the infrastructure in the vehicle.

5.4.4 RISEN deployed in a secure cloud environment

In this setting, the RISEN sensors are transported and installed at the crime scene. The 3DA-CSI System component is deployed in a secure cloud environment, accessible to all RISEN sensors via a virtual private network (VPN) provider or directly over the Internet, with adequate security mechanisms in place.

A local wireless network is established to connect all RISEN components. Where needed, network repeaters are installed to ensure network coverage. The local network connects to an Internet Service Provider (ISP) delivering connectivity to the remote 3DA-CSI System component.

The RISEN sensors send measurements to the 3DA-CSI System component. The investigator accesses the 3DA-CSI System locally using a computer in the vehicle or at the crime scene. All information is stored in the 3DA-CSI System component.

In this setting, it is possible to remotely access information from the 3DA-CSI System (e.g., request assessment from an expert not located at the crime scene).

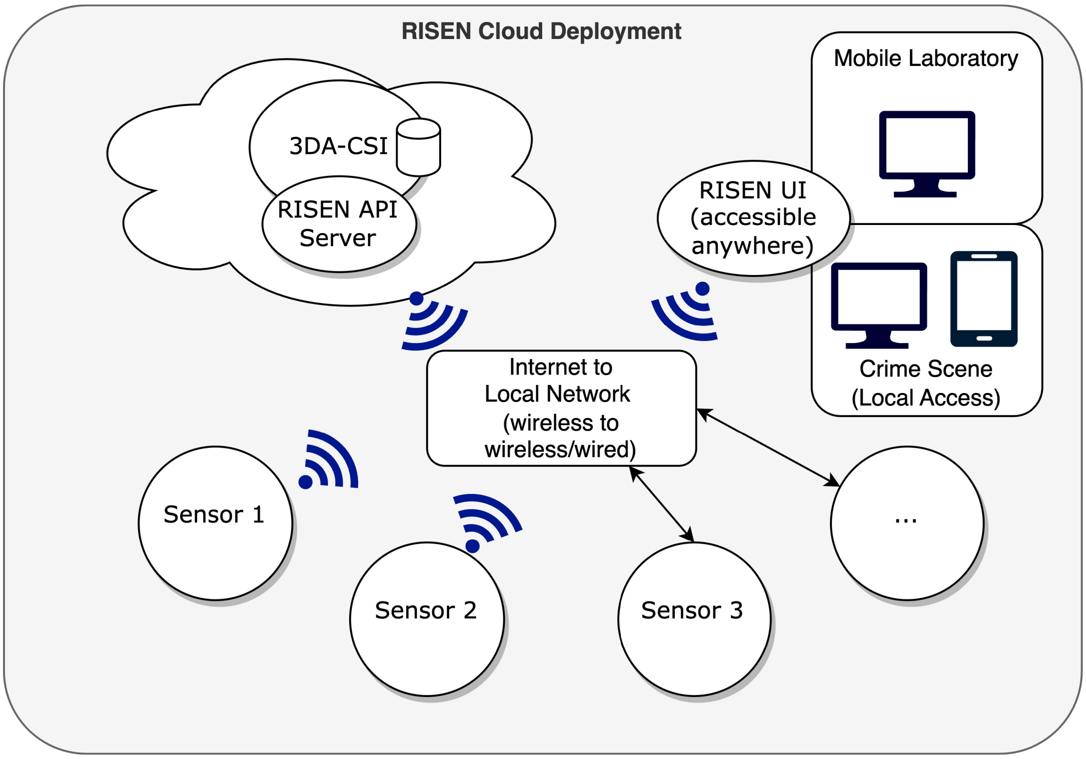

RISEN Cloud Deployment
3DA-CSI
RISEN API Server
Mobile Laboratory
RISEN UI (accessible anywhere)
Crime Scene (Local Access)
Internet to Local Network (wireless to wireless/wired)
Sensor 1
Sensor 2
Sensor 3
...

Figure 4 — Cloud Deployment

This setting is advantageous for situations where network connectivity with a central 3DA-CSI System is possible and desired. It also takes advantage of advanced computing capabilities provided by a stable infrastructure.

However, the setting requires continuous broadband network connectivity, a requirement that is not present in all crime scenes. Moreover, security concerns might arise concerning assurance of data confidentiality. The fact that the 3DA-CSI System is accessible over the Internet (even via a VPN) may bring additional security risks that might not be acceptable by many LEAs.

6 RISEN Sensor API Data Model

6.1 General

The RISEN Sensor API data model defines how sensor data is represented in the RISEN System. The model is represented by several entities, structured in the following categories:

- Authentication entities that deal with authentication and permissions in the RISEN System.
- Sensor entities that deal with the representation of sensors and sensor-generated data.
- Ancillary entities that contain additional information related to several entities.

Entities' records are managed based on unique identifiers generated by the RISEN System. In order for the RISEN System to function properly, it is mandatory to ensure the identifiers' uniqueness, from a system perspective. The RISEN System can generate and manage identifiers assuring they are unique to each specific record. Alternatively, the RISEN System has been designed to allow the API client to use its own identifiers, where applicable (e.g., a sensor identifier generated by the manufacturer). In such case, however, it becomes the responsibility of the API client to ensure the uniqueness of the identifiers.

It is noted as general rules:

- The RISEN System generates and manages its own identifiers via the 'id' field. This field is internal to the RISEN System and is not shared with API clients.
- The API client may (and should) generate its own identifiers using the 'external_id' field. Subsequent access and operations to related entities can be done using this field.
- If the API client does not generate identifiers in the 'external_id', the RISEN System shall generate them and the API client must be capable to operate with the RISEN generated identifiers.

6.2 Authentication entities

These entities deal with defining access rights, permissions and privileges in the RISEN System (Groups and Permission entities), as well as users (User entity).

This document will not present details related with authentication and authorisation specifications, except for authentication operations concerning login and logout.

Entity	Fields	Description
User	id: unique identifier username: unique identifier that identifies a specific user (string) password: string email_address: email active: boolean (indicates if the user is active)	Represents a user of the RISEN System.

6.3 Sensor entities

6.3.1 General

Sensor entities deal with the representation of sensors information (e.g., operation and battery status) and sensor events (e.g., measurements).

Figure 5 depicts the defined Sensor entities that are described next.

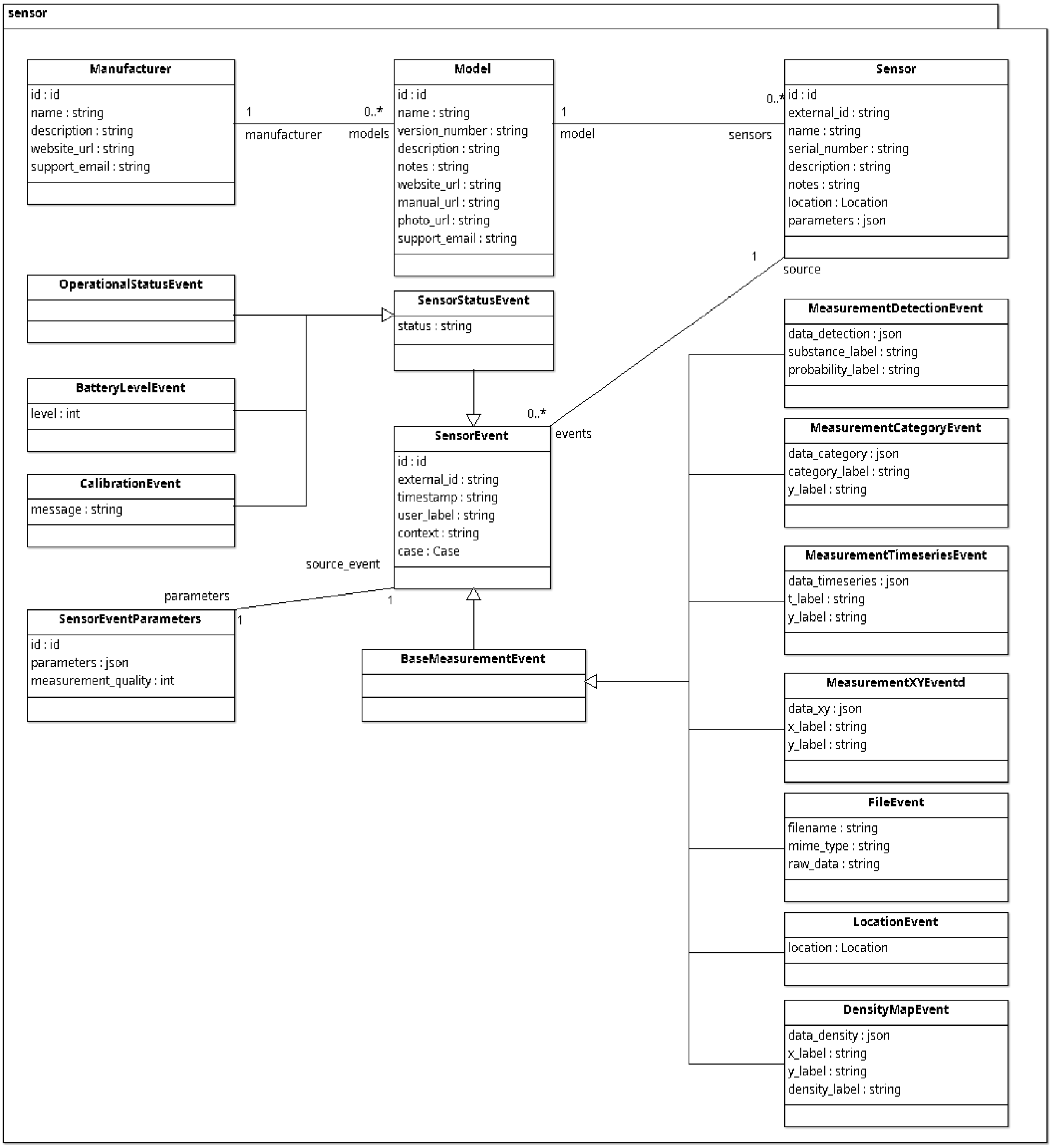


Figure 5 — Sensor Information Entities

6.3.2 RISEN sensor event entities

These entities represent data (or events) generated by the RISEN sensors. It comprises a central entity named "SensorEvent" that provides fields common to sensor data, such as timestamp and context. A sensor can generate multiple SensorEvents.

Several entities are created representing specific sensor data related with, for example, operational status, battery level, calibration and measurements. In practice, these entities "extend" the SensorEvent, inheriting its fields and adding new ones to address needed specificities.

Entity	**Fields**	**Description**
SensorEvent	id: unique identifier (auto) external_id: string (may refer to an identifier external to RISEN) timestamp: date and time in ISO 8601 format (e.g., 2021-01-01T12:05:10+00:00) user_label : string location: LocationEvent context: string (may contain additional information in arbitrary format) case_event: link to CaseEvent source: link to Sensor	Describes common fields in a sensor data event.
SensorEventParameter	id: unique identifier (auto) parameters: json measurement_quality: int (0: none, 1: lowest, 5: highest)	Describes a sensor's parameters related with a specific measurement (sensor event). Since parameters are sensor specific, this field contains a json structure with an arbitrary structure.
SensorStatusEvent	status: string (generic representation of a sensor status. See its subclasses)	
OperationalStatusEvent	status: sets as enumeration to: Offline, Idle, Warm Up, Calibrating, Standby, Acquiring(Active), Error, Unknown	Describes a sensor's operational status.
BatteryLevelEvent	status: sets as enumeration to: On Battery, On Power, Unknown level: double [0.0 … 1.0]	Describes a sensor's battery status.
CalibrationEvent	status: sets as enumeration to: Not calibrated, Calibrated, Need to recalibrate, Unknown message: string (max. 3000 chars)	Describes a sensor's calibration status.
MeasurementDetectionEvent	data_detection: list of detected substances with respective probability	Describes a sensor's measurement detection.
MeasurementCategoryEvent	category_data: data for a category (data series) chart, allowing multiple ordinate values.	Describes a sensor measurement in multiple dimensions. Format based on C3 (https://c3js.org/) or HDF.

Entity	Fields	Description
MeasurementXYEvent	xy_data: data for a X-Y chart, allowing multiple ordinate values.	Describes a sensor measurement in two dimensions, where the x-axis refers to a scale. Format based on C3 (https://c3js.org/).
MeasurementTimeseriesEvent	timeseries_data: data for a timeseries chart, allowing multiple ordinate values.	Describes a sensor measurement in timeseries. Format based on C3 (https://c3js.org/).
DensityMapEvent	data_density: 3 dimensional matrix in json array format.	Describes a sensor measurement as a density map. Format based on Plotly (https://plotly.com/javascript/heatmaps/)
FileEvent	id: unique identifier (auto) filename: string representing name of the file associated with the file (optional) mime_type: as per IANA IETF RFC 6838 (see https://datatracker.ietf.org/doc/html/rfc6838) Examples: application/pdf ; model/vrml raw_data: file data (in base64)	Generic representation of a file. Note that it can also represent images and 3D models.

6.4 Ancillary entities

These represent additional information used by other entities. These include a location entity (with global and relative coordinates, including orientation), and a log message entity (allowing recording a message).

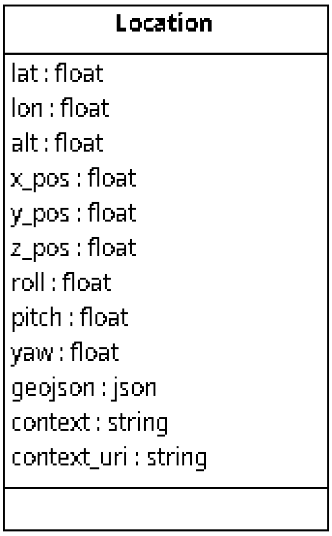


Figure 6 — Ancillary Entities

Entity	Fields	Description
Location	id: unique identifier (auto) timestamp: date and time in UTC (ISO8601 format) external_id: string (may refer to an identifier external to RISEN) Global Coordinates: latitude (number), longitude (number), altitude (number) Local (relative) Coordinates: x (number), y (number), z (number), roll (number), pitch (number), yaw (number)	Describes the location of events or entities. Uses global and local coordinates.
LogMessage	id: unique identifier (auto) timestamp: date and time in UTC (ISO8601 format) level: enumeration: debug, information, warning, error, critical description: string (max. 3000 chars) source_id: integer (used to refer to link an existing entity)	Generic class used to log messages from sensors (or any other sources).

7 RISEN Sensor API Specification

7.1 General

The RISEN Sensor API is specified using the OpenAPI Specification (OAS) [ref-OAS]. Documentation can be generated directly from source-code, reducing time to develop and produce documents, while also keeping source code and documentation synchronised.

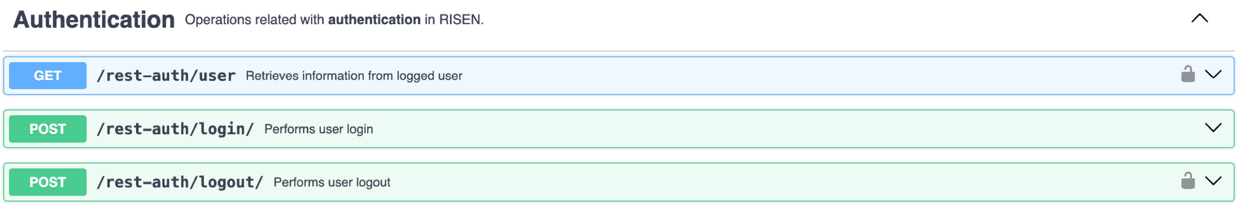
The RISEN Sensor API follows a similar structure as the sensor categories defined in section 4, specifically:

- Authentication: representing interfaces related with authentication.
- Sensor Events: representing interfaces related with sensors status and measurements.

7.2 Authentication

This provides interfaces related with authentication in the RISEN System, including login, logout and retrieve information related with the logged user. The interfaces support GET and POST operations.

A user is authenticated by performing a “login” sending valid credentials information.


Authentication Operations related with **authentication** in RISEN.

GET /rest-auth/user Retrieves information from logged user

POST /rest-auth/login/ Performs user login

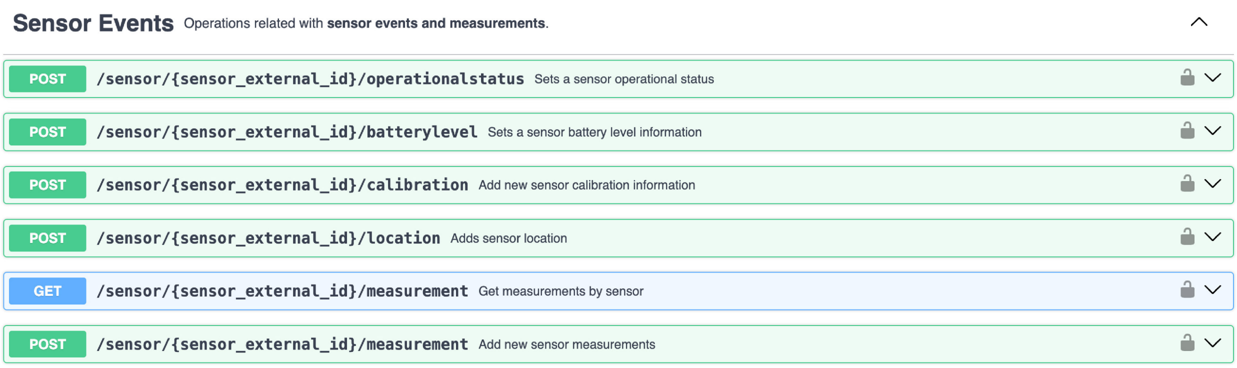
POST /rest-auth/logout/ Performs user logout

Figure 7 — Authentication

7.3 Sensor events

7.3.1 General

This provides interfaces related with sensor events. It can be used to send sensor measurements and status information. It supports POST operations.



Sensor Events Operations related with **sensor events and measurements**.

POST /sensor/{sensor_external_id}/operationalstatus Sets a sensor operational status

POST /sensor/{sensor_external_id}/batterylevel Sets a sensor battery level information

POST /sensor/{sensor_external_id}/calibration Add new sensor calibration information

POST /sensor/{sensor_external_id}/location Adds sensor location

GET /sensor/{sensor_external_id}/measurement Get measurements by sensor

POST /sensor/{sensor_external_id}/measurement Add new sensor measurements

Figure 8 — Sensor Events

Each sensor event interface has a specific associated entity. It is required to provide the sensor identifier (i.e., sensor_external_id) as part of the POST operation.

Entities used in sensor events are presented in the following paragraphs.

7.3.2 Operational status

```
{
        "external_id": "string",
        "timestamp": "2021-01-01T12:05:10+00:00",
        "user_label": "string",
        "context": "string",
        "status": "Offline"
}
```

NOTE 1 "status" is the enumeration: [Offline, Idle, WarmUp, Calibrating, Standby, Acquiring, Error, Unknown].

7.3.3 Battery level

```
{
        "external_id": "string",
        "timestamp": "2021-01-01T12:05:10+00:00",
        "user_label": "string",
        "context": "string",
        "status": "On Battery",
        "level": 0
}
```

NOTE 1 "status" is the enumeration: [On Battery, On Power, Unknown].

NOTE 2 In case the sensor does not provide battery level information, the field "level" should not be included or should be filled with "null".

7.3.4 Calibration level

```
{
       "external_id": "string",
       "timestamp": "2021-01-01T12:05:10+00:00",
       "user_label": "string",
       "context": "string",
       "status": "Calibrated",
       "message": "Calibration completed OK."
}
```

NOTE 1 "status" is the enumeration: [Not Calibrated, Calibrated, Need to Calibrate, Unknown].

NOTE 2 "message" is an optional arbitrary text provided by the sensor.

7.3.5 Location

```
{
       "timestamp": "2021-01-01T12:05:10+00:00",
       "latitude": 0,
       "longitude": 0,
       "altitude": 0,
       "x_pos": 0,
       "y_pos": 0,
       "z_pos": 0,
       "roll": 0,
       "pitch": 0,
       "yaw": 0
}
```

NOTE Some fields might be omitted if their value is not known.

7.3.6 Measurement

The measurement endpoint can be used to send sensor measurements, which can contain one or more measurement events. Moreover, information concerning the applicable sensor parameters should also be included under “parameters”.

```
       "measurement": {
           "user_label": "Text",
           "parameters": json,
           "data_detection": [
             {
               "title": "Title",
               "description": "Description",
               "substance_label": "Label",
              "probability_label": "Probability",
               "note": "Note.",
               "substances": [
                 {
                   "substance": "Formaldehyde",
                   "code": "CAS:50-00-0",
                   "probability": 0.6
                 },
                 {
                   "substance": "Cocaine",
                   "code": "CAS:50-36-2",
                   "probability": 0.3
                 }
               ]
             }
```

```
],
"data_xy": [
  {
    "title": "Title",
   "description": "Description",
    "x_label": "Time (seconds)",
   "y_label": "Intensity (unit...)",
    "x_accuracy": 0.1,
    "y_accuracy": 0.1,
    "x": "x",
    "columns":[
       [ "x", x_data ],
       [ "Measurement A", A_data ],
       [ "Measurement B", B_data ],
      ]
  }
],
"data_timeseries": [
  {
   "title": "Title",
   "description": "Description",
   "t_label": "Time (seconds)",
   "y_label": "Intensity (units...)",
   "t_accuracy": 0.1,
   "y_accuracy": 0.1,
   "t": "t",
   "columns": [
      [ "t", t_data ],
      [ "Time window 1", time1_data ],
      [ "Time window 2", time2_data ]
    ]
  }
],
"data_category": [
  {
   "title": "Category",
   "description": "Results per category",
   "category_label": "Substances",
   "y_label": "Result",
    "category": "category",
    "category_accuracy": 0.1,
    "y_accuracy": 0.1,
    "columns": [
      [ "category", "A", "B" ],
      [ "data1", data1 ],
      [ "data2", data2 ]
    ]
  }
],
"data_densitymap": [
{
  "title": "Density Map",
  "description": "Description",
  "columns": [
    [ entry_1_with_3_values ],
    [ entry_2_with_3_values ],
   ...
  ],
  "x_label": "X label (unit)",
  "y_label": "Y Label (unit)",
  "density_label": "Density Label"
  }
```

```
            ],
            "location": {
              "latitude": 38.71667,
              "longitude": -9.13333,
              "altitude": 0.0,
              "x_pos": 0,
              "y_pos": 0,
              "z_pos": 0,
              "roll": 0,
              "pitch": 0,
              "yaw": 0
            }
        }
```

NOTE 1 A measurement record can contain different and multiple types of data (e.g., one data_detection entry and two data_timeseries entries).

NOTE 2 Fields “Title” and “Description” should be used to provide specific information concerning the provided data, such as “Detection”, “Spectrum” and “Chromatogram”.

NOTE 3 Whenever applicable, in a metric label it is recommended to include unit information inside brackets. For example “Time (seconds)”.

NOTE 4 For classification purposes it is recommended to use international nomenclature and codes for chemical substances and biological agents. For example, substance “Acetone” would include its international code “cas:67-64-1” or “ec:200-662-2”. The following sources were used in RISEN:

— EC ECHA: https://echa.europa.eu/information-on-chemicals/ec-inventory

— GESTIS Substance Database: http://www.dguv.de/ifa/gestis-database

— GESTIS list of biological agents: http://www.dguv.de/ifa/gestis-biological-agents

— CAS Registry: https://commonchemistry.cas.org

8 Conclusion

As part of the RISEN Action and with the support of the German Institute for Standardization (DIN), the CEN Workshop Agreement was launched to produce the document “CBRNe SENSOR API — Network protocols, data formats and interfaces”. This document defines the application programming interface (API) as a mechanism that allows sensors, as well as software components in general, to operate with a compliant remote server in a modular way. By following the RISEN API specifications, any authorised component can seamlessly connect to and exchange information with the RISEN System.

This document started by presenting the API overall architecture and possible deployment options, thus setting the frame to describe proposed data models and detailed specifications, which can be followed by sensor manufactures to produce compliant products.

The API was designed to be modular, extensible and interoperable. For its specification, widely adopted and open Internet-based standards and technologies, thus facilitating its incorporation in sensor systems.

It is the intent of this API to evolve over future versions, including the incorporation of new interfaces, new data types, domain taxonomy and explore additional technologies, like the use of message brokers based on the publish-subscribe paradigm to enable event-driven processing.

Bibliography

[1] CHAMPOD C. In: *Encyclopedia of Forensic Sciences*. (SIEGEL J.A., SAUKKO P.J., eds.). Academic Press, Waltham, 2013, pp. 303–9.

[2] ENSFI, Scenes of Crime Examination Best Practice Manual.

[3] ENFSI-BPM-SOC-01 Version 02, February 2022. https://enfsi.eu/wp-content/uploads/2022/02/BPM-SOC-01-v.20220214_final_v2.pdf

[4] NFSTC. Crime Scene Investigation - A Guide for Law Enforcement. September 2013. https://www.nist.gov/system/files/documents/forensics/Crime-Scene-Investigation.pdf

[5] ENSFI, BEST PRACTICE MANUAL FOR THE FORENSIC COMPARISON OF SOIL TRACES. ENFSI-BPM-APS-02 Version 1 – December 2019. https://enfsi.eu/wp-content/uploads/2017/06/ENFSI_BPM_APST_Soil_Examination-vs1.0.pdf

[6] ENSFI. Best Practice Manual for controlled drug analysis. ENSFI-DWG-CDA-001 Version 01, January 2020. https://enfsi.eu/wp-content/uploads/2017/06/BPM-Control-drug-Analysis-final-version.-21-02-2020.pdf

[7] HILDEBRANDT M., KILTZ S., DITTMANN J. "Digitized forensics: retaining a link between physical and digital crime scene traces using QR-codes". Proc. SPIE 8667, Multimedia Content and Mobile Devices. March 2013. https://www.spiedigitallibrary.org/conference-proceedings-of-spie/8667/1/Digitized-forensics--retaining-a-link-between-physical-and-digital/10.1117/12.2004548.short?SSO=1

[8] ENCYCLOPÆDIA BRITANNICA. https://www.britannica.com/topic/evidence-law

[9] NIJ. Landscape Study on 3D Crime Scene Scanning Devices. January 2016. https://www.rieglusa.com/pdf/landscape-study-on-3d-crime-scene-scanning-devices.pdf

[10] MARGOT P. Forensic science on trial - What is the law of the land? *Aust. J. Forensic Sci.* 2011, **43** pp. 2–3, 89–103

[11] OPENAPI INITIATIVE. OAS Open API Specifications v3.0.3. February 2020. https://spec.openapis.org/oas/v3.0.3#:~:text=The%20OpenAPI%20Specification%20(OAS)%20defines,or%20inspection%20of%20network%20traffic.

[12] IETF RFC 72311, Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. *Internet Engineering Task Force*. Available at: https://datatracker.ietf.org/doc/html/rfc7231

[13] RISEN Consortium. D4.3 RISEN Sensor API Definition Document v1.0. Dated: 30-08-2021

[14] ISO 21043-1:2018, *Forensic sciences — Part 1: Terms and definitions*

[15] ISO 21043-2:2018, *Forensic sciences — Part 2: Recognition, recording, collecting, transport and storage of items*

[16] ISO/CD 21043-3, *Forensic Sciences — Part 3: Analysis*[1]

[17] ISO/CD 21043-4, *Forensic Sciences — Part 4: Interpretation*[1]

[18] ISO/CD 21043-5, *Forensic Sciences — Part 5: Reporting*[1]

[19] ISO/IEC 30128:2014, *Information technology — Sensor networks — Generic Sensor Network Application Interface*

[20] ISO/IEC series 29182, *Information technology — Sensor networks: Sensor Network Reference Architecture (SNRA)*