Accordo ASI - INAF n. 2020-35-HH.O

Agenzia Spaziale Italiana

INAF ISTITUTO NAZIONALE DI ASTROFISICA NATIONAL INSTITUTE FOR ASTROPHYSICS

23/11/2023 **ESWW 2023**

CÆSAR
COMPREHENSIVE SPACE WEATHER STUDIES FOR THE ASPIS PROTOTYPE REALIZATION

ProSpecT

Simplifying Product Description and Metadata Generation for the CAESAR Project

V. Formato on behalf of the CAESAR team

Project Prime:

INAF ISTITUTO NAZIONALE DI ASTROFISICA NATIONAL INSTITUTE FOR ASTROPHYSICS

Project Partners:

INFN Istituto Nazionale di Fisica Nucleare

ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA

TOR VERGATA UNIVERSITÀ DEGLI STUDI DI ROMA

unipg UNIVERSITÀ DEGLI STUDI DI PERUGIA

Università di Genova

UNIVERSITÀ DEGLI STUDI DELL'AQUILA

UNIVERSITÀ DELLA CALABRIA

Università di Catania

UNIVERSITÀ DI TRENTO

CAESAR

(Comprehensive spAce wEather Studies for the ASPIS prototype Realization)

The project goals are:

- Encompass all the relevant aspects of Space Weather science
- Implement a prototype for **ASPIS** (*ASI SPace Weather InfraStructure*): the scientific data center for Space Weather of the Italian Space Agency (**ASI**)

The project involves a large part of the **Space Weather Italian community** bringing together 10 Italian institutions as partners:

- 3 national research institutes
  - **INAF, INGV, INFN**
- 7 universities
  - **Tor Vergata (Rome II), Perugia, Genoa, L'Aquila, Calabria, Catania, Trento**

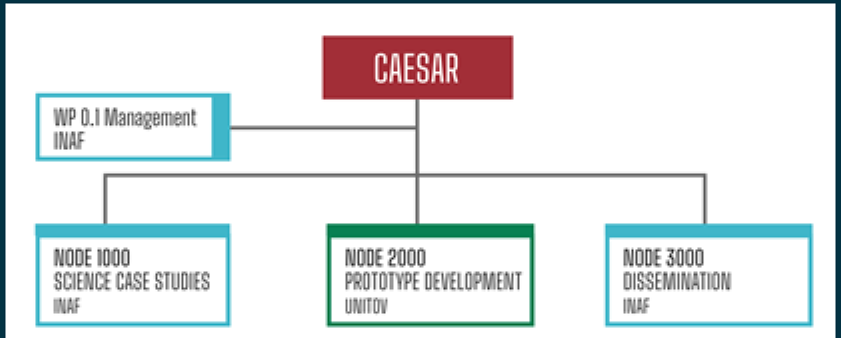
CAESAR will aim to encompass the whole chain of phenomena from the Sun to the Earth up to planetary environments, selecting a number of well-observed "target Space Weather events" for detailed and comprehensive studies in order to showcase the proposed approach.

CAESAR

(Comprehensive spAce wEather Studies for the ASPIS prototype Realization)

The CAESAR project is split into three main “nodes”:

- Node 1000
  - Science cases: *collects all relevant use cases and data products to be included in the project*
- Node 2000
  - Prototype development: *study and design the actual implementation of the platform*
- Node 3000
  - Dissemination: *outreach activities and raise awareness on the CAESAR/ASPIS projects*



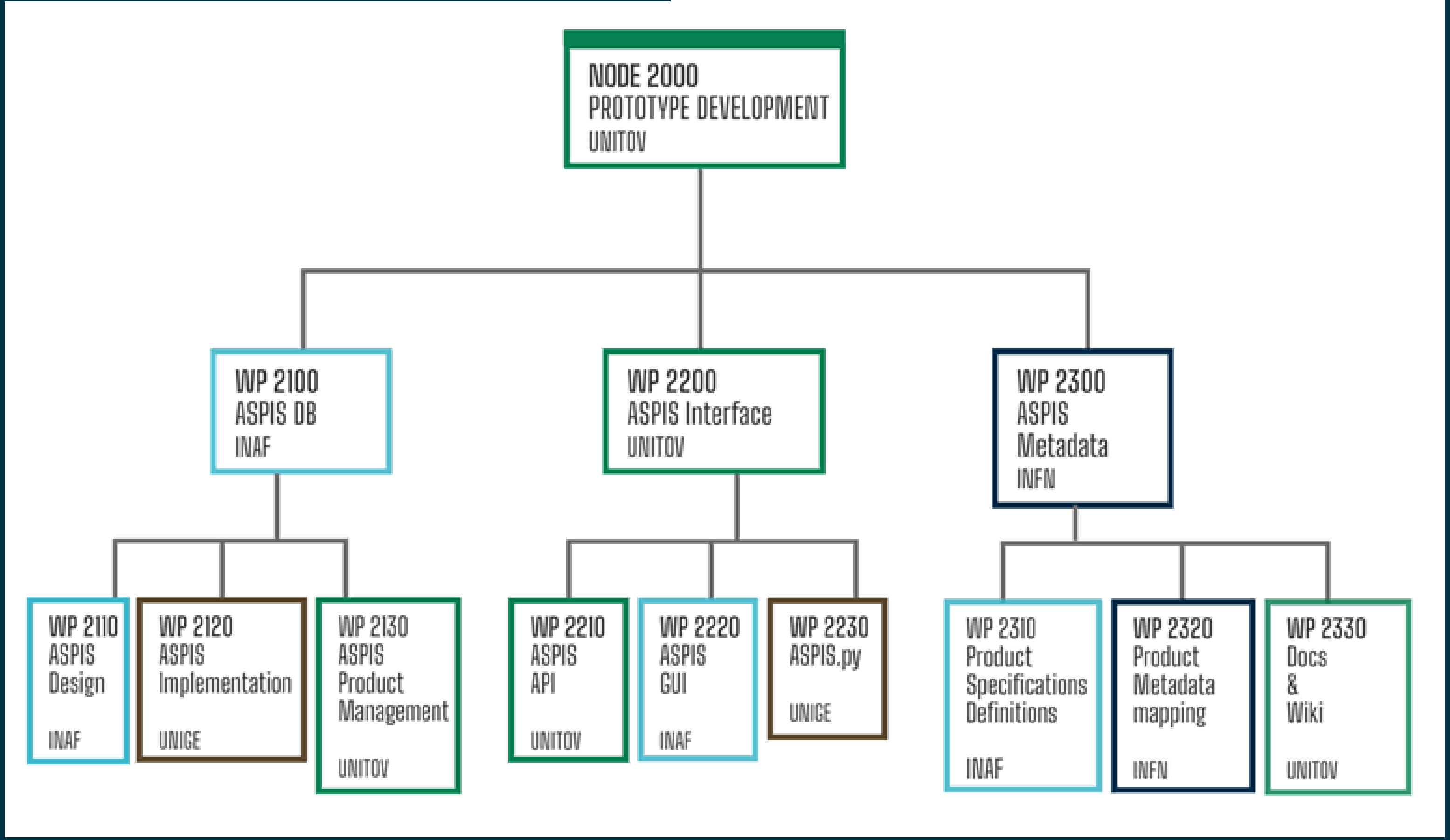
CAESAR

WP 0.1 Management
INAF

NODE 1000
SCIENCE CASE STUDIES
INAF

NODE 2000
PROTOTYPE DEVELOPMENT
UNITOV

NODE 3000
DISSEMINATION
INAF



NODE 2000
PROTOTYPE DEVELOPMENT
UNITOV

WP 2100
ASPIS DB
INAF

WP 2200
ASPIS Interface
UNITOV

WP 2300
ASPIS
Metadata
INFN

WP 2110
ASPIS
Design
INAF

WP 2120
ASPIS
Implementation
UNICE

WP 2130
ASPIS
Product
Management
UNITOV

WP 2210
ASPIS
API
UNITOV

WP 2220
ASPIS
GUI
INAF

WP 2230
ASPIS.py
UNICE

WP 2310
Product
Specifications
Definitions
INAF

WP 2320
Product
Metadata
mapping
INFN

WP 2330
Docs
&
Wiki
UNITOV

CAESAR *(Node 2000)*

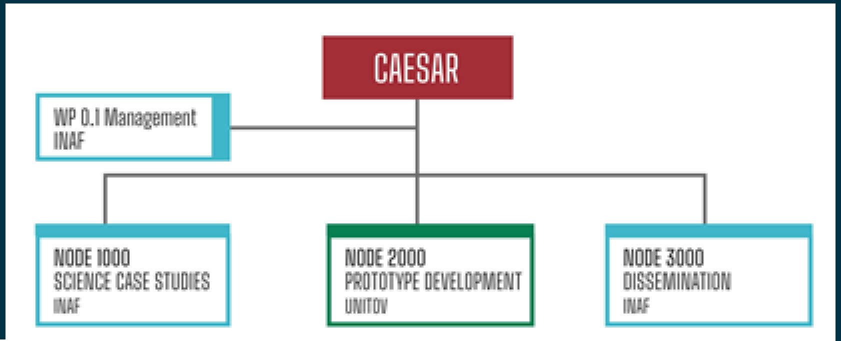
(Comprehensive spAce wEather Studies for the ASPIS prototype Realization)

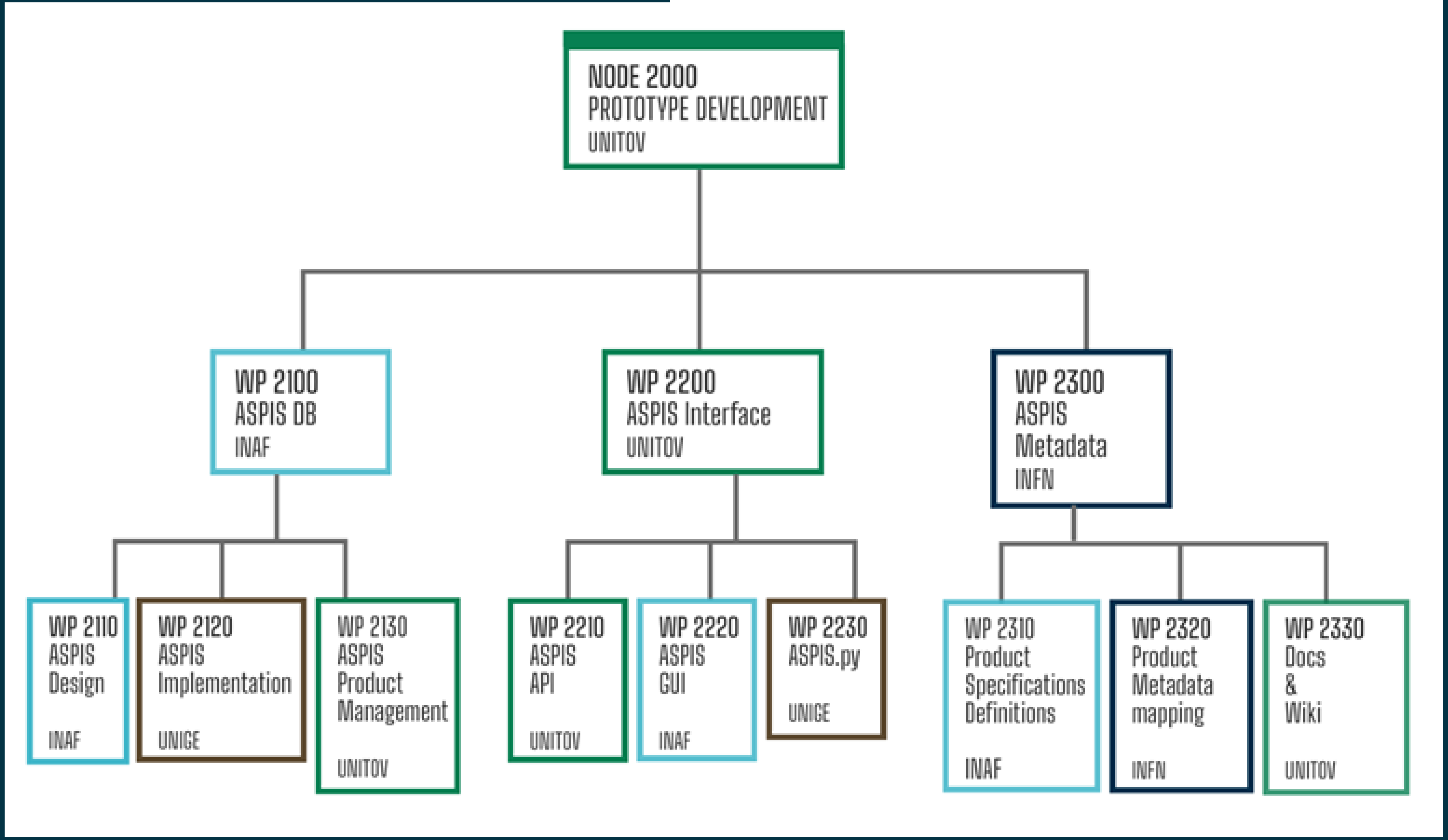
On the technical side the prototype implementation will be based on three main pillars:

- Database [WP2100]
- Interface [WP2200]
- Metadata mapping [WP2300]

The ASPIS DB will contain mainly proprietary/co-proprietary products, with their relative data policy in a homogeneous, standardized collection of resources.

Other important external data shall be accessed through links to existent archives.



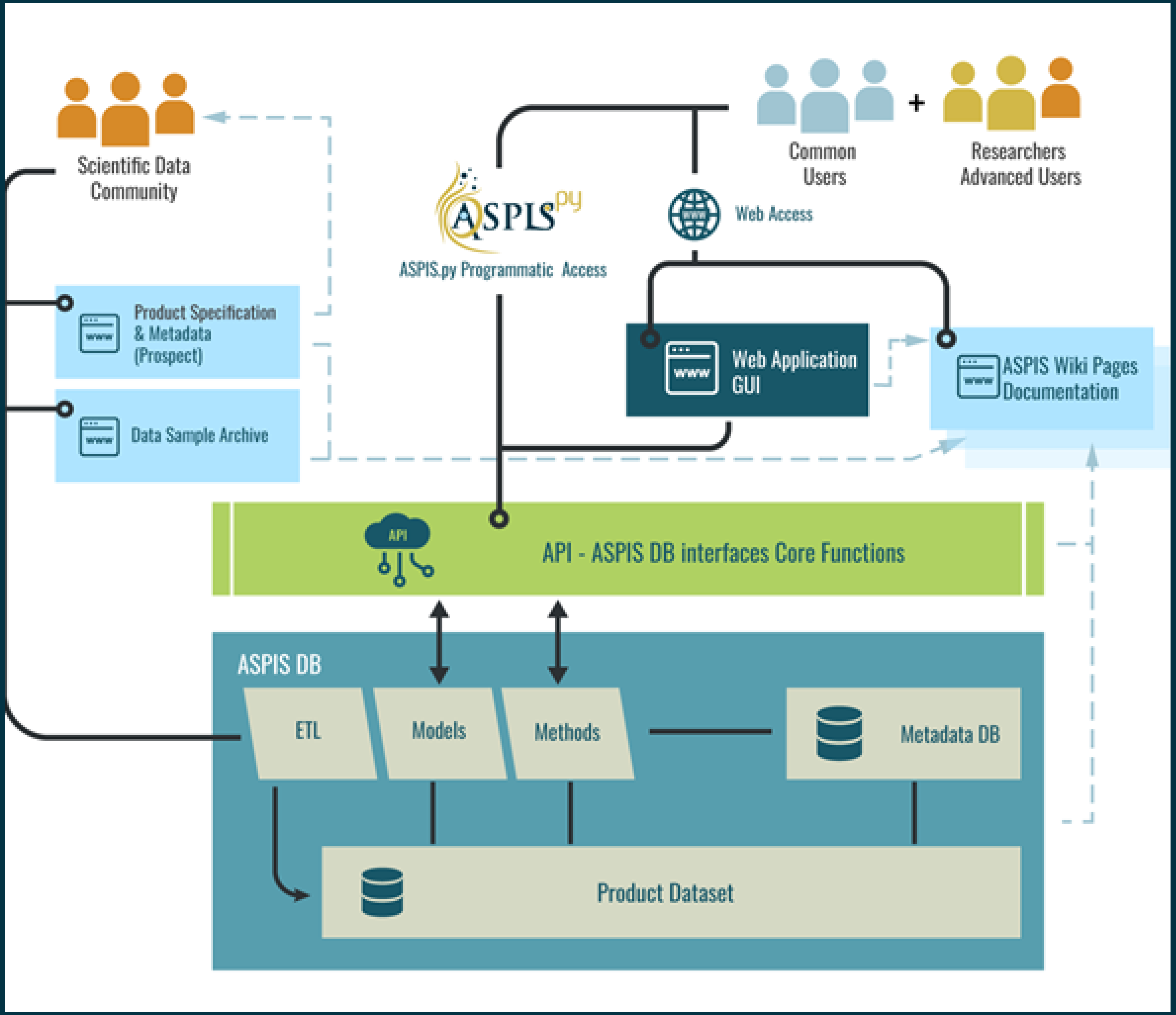


CAESAR *(Node 2000)*

(Comprehensive spAce wEather Studies for the ASPIS prototype Realization)

High Level Working Packages are responsible for

- WP2100
  - Designing the DB
    (database in its general meaning)
  - Implement it
    - Empty → preliminary→ final prototype
  - Ingest product data/models
  - Incorporate models/tools
- WP2200
  - Define the archive’s API
  - Provide a GUI (available to all user)
  - Develop ASPIS.py (for advanced researchers)
- WP2300
  - Templating product descriptions
    (and collect them from Node 1000)
  - Map metadata content and formats
    (for internal/external usage)
  - Document all the activities and processes



Scientific Data Community
Common Users
+
Researchers Advanced Users
ASPIS py
ASPIS.py Programmatic Access
Web Access
Product Specification & Metadata (Prospect)
Data Sample Archive
Web Application GUI
ASPIS Wiki Pages Documentation
API
API - ASPIS DB interfaces Core Functions
ASPIS DB
ETL
Models
Methods
Metadata DB
Product Dataset

Choosing a data format

Before collecting product specifications from users in Node 1000 we needed to choose a unique data format to store them

Several factors were taken into account:

- We have a high number of products, coming from several physics domains and each with its own custom data format
- It is very unlikely to have common fields or variable names between different products

Choosing a data format

Before collecting product specifications from users in Node 1000 we needed to choose a unique data format to store them

Several factors were taken into account:

- We have a high number of products, coming from several physics domains and each with its own custom data format
- It is very unlikely to have common fields or variable names between different products
- Data structures can be arbitrarily nested or complex, with variable size or content

Choosing a data format

Before collecting product specifications from users in Node 1000 we needed to choose a unique data format to store them

Several factors were taken into account:

- We have a high number of products, coming from several physics domains and each with its own custom data format
- It is very unlikely to have common fields or variable names between different products
- Data structures can be arbitrarily nested or complex, with variable size or content

This pretty much excludes a tabular data format

Choosing a data format

The choice fell on the **JSON** (JavaScript Object Notation) format

It is represented as a collection of key-value pairs, where the value can store several data types, including other JSON documents (which allows arbitrary nesting of data)

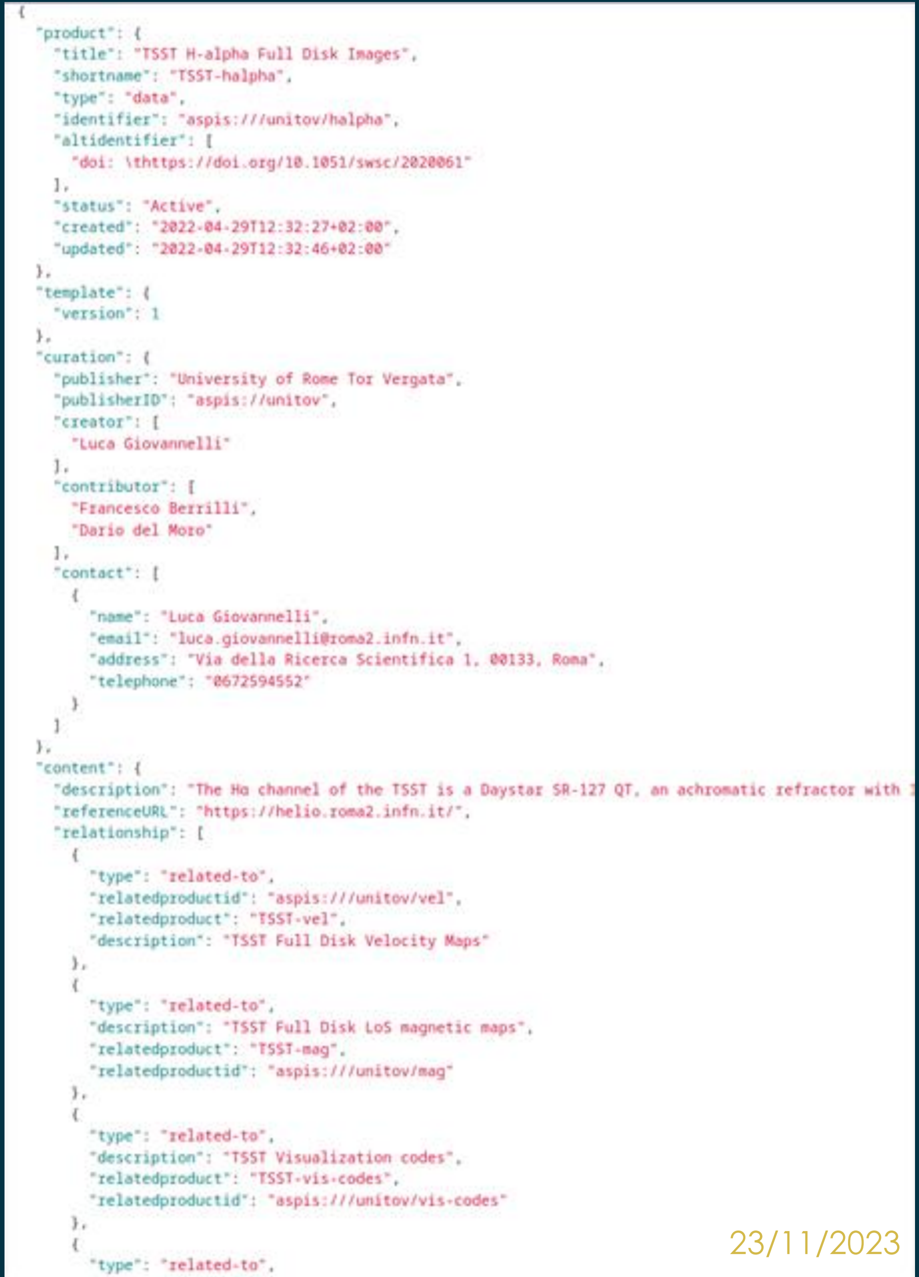
It's both very flexible and powerful, and it allows to store data as strings, numbers, sequence of other values (arrays), or other JSON documents.
It was designed to be human-readable, since each value in the document is effectively "named" by its key.

```json
{
  "firstName": "John",
  "lastName": "Smith",
  "isAlive": true,
  "age": 27,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021-3100"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "office",
      "number": "646 555-4567"
    }
  ],
  "children": [
      "Catherine",
      "Thomas",
      "Trevor"
  ],
  "spouse": null
}
```

Metadata Design

With a data format chosen, we then proceeded to lay down a set of crucial metadata that would help catalogue and efficiently search among products including:

- A title, shortname and unique identifier, and a type determining if the product is a dataset or a scientific model
- A "status" indicating if the product is actively maintained and updated
- Info about the product "curator" institution, which includes names of contributors and a contact person
- An accurate and concise description of the product data and its layout
- A complete set of relationships with other products
- ...and much more



```
{
  "product": {
    "title": "TSST H-alpha Full Disk Images",
    "shortname": "TSST-halpha",
    "type": "data",
    "identifier": "aspis:///unitov/halpha",
    "altidentifier": [
      "doi: \thttps://doi.org/10.1051/swsc/2020061"
    ],
    "status": "Active",
    "created": "2022-04-29T12:32:27+02:00",
    "updated": "2022-04-29T12:32:46+02:00"
  },
  "template": {
    "version": 1
  },
  "curation": {
    "publisher": "University of Rome Tor Vergata",
    "publisherID": "aspis://unitov",
    "creator": [
      "Luca Giovannelli"
    ],
    "contributor": [
      "Francesco Berrilli",
      "Dario del Moro"
    ],
    "contact": [
      {
        "name": "Luca Giovannelli",
        "email": "luca.giovannelli@roma2.infn.it",
        "address": "Via della Ricerca Scientifica 1, 00133, Roma",
        "telephone": "0672594552"
      }
    ]
  },
  "content": {
    "description": "The Hα channel of the TSST is a Daystar SR-127 QT, an achromatic refractor with 
    "referenceURL": "https://helio.roma2.infn.it/",
    "relationship": [
      {
        "type": "related-to",
        "relatedproductid": "aspis:///unitov/vel",
        "relatedproduct": "TSST-vel",
        "description": "TSST Full Disk Velocity Maps"
      },
      {
        "type": "related-to",
        "description": "TSST Full Disk LoS magnetic maps",
        "relatedproduct": "TSST-mag",
        "relatedproductid": "aspis:///unitov/mag"
      },
      {
        "type": "related-to",
        "description": "TSST Visualization codes",
        "relatedproduct": "TSST-vis-codes",
        "relatedproductid": "aspis:///unitov/vis-codes"
      },
      {
        "type": "related-to",
```

Pros / Cons

JSON files are easy to inspect by humans (useful for Node 2000 people during the prototype development)

JSON files are easy to handle by computers, allowing Node 2000 to easily automate part of their tasks (e.g. generation of wiki entry for each product, DB schema design, etc...)

JSON can be difficult to write (and time consuming) especially for first-time users, which includes most if not all the data providers in Node 1000

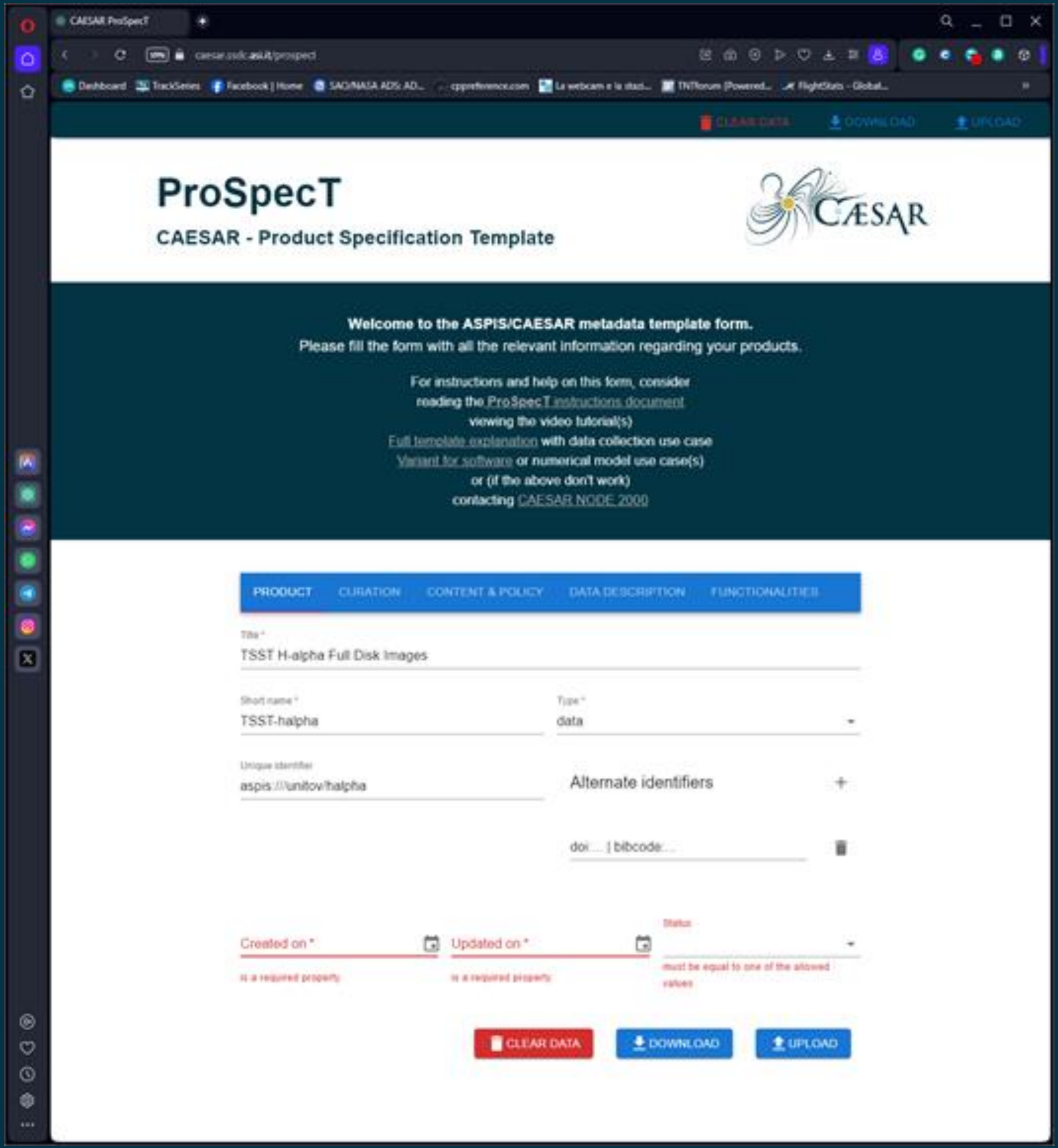
In order to make this process as streamlined and intuitive as possible we developed a dedicated web application that Node 1000 users could use to fill in their product metadata and save them directly in JSON format

ProSpecT

The **ProSpecT** (Product Specification Template) project is based on the JSONForms web framework (https://jsonforms.io)

It allows users to input all the product metadata in pre-determined fields and provides some basic form of input validation to make sure the metadata are in the right form

Metadata are divided in sections to ease navigation and to not overwhelm the user with dozens of fields on the same page



ProSpecT

CAESAR - Product Specification Template

Welcome to the ASPIS/CAESAR metadata template form.
Please fill the form with all the relevant information regarding your products.

For instructions and help on this form, consider
reading the ProSpecT instructions document
viewing the video tutorial(s)
Full template explanation with data collection use case
Variant for software or numerical model use case(s)
or (if the above don't work)
contacting CAESAR NODE 2000

PRODUCT | CURATION | CONTENT & POLICY | DATA DESCRIPTION | FUNCTIONALITIES

Title *: TSST H-alpha Full Disk Images

Short name *: TSST-halpha

Type *: data

Unique identifier: aspis:///unitov/halpha

Alternate identifiers

doi:... | bibcode:...

Created on *
is a required property

Updated on *
is a required property

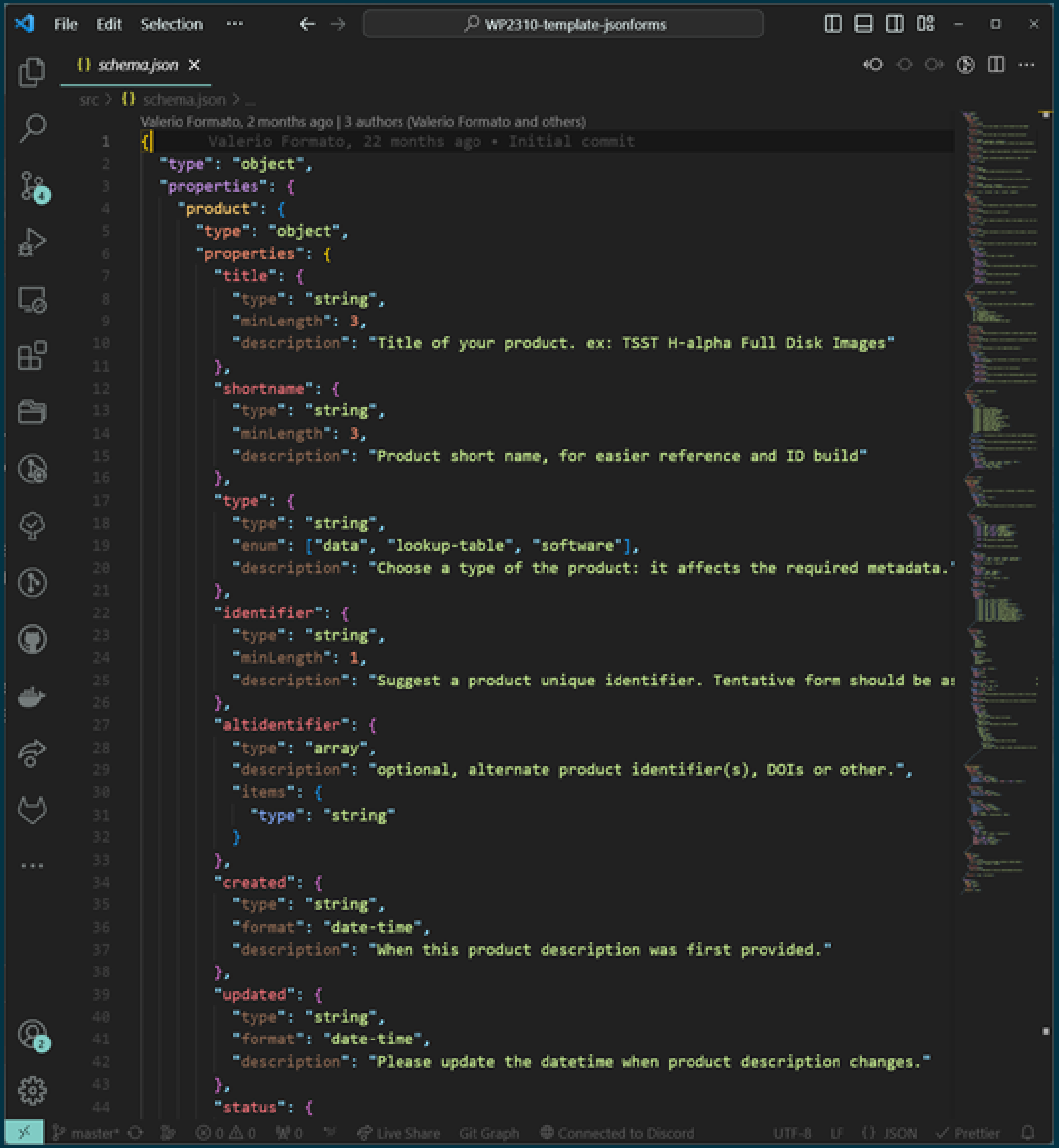
Status
must be equal to one of the allowed values

CLEAR DATA | DOWNLOAD | UPLOAD

ProSpecT

The data in the forms will be stored in a JSON object.

JSONForms will use a special “schema.json” file to infer the final layout of the output JSON forms content.


File
Edit
Selection
WP2310-template-jsonforms
schema.json
src > schema.json > ...
Valerio Formato, 2 months ago | 3 authors (Valerio Formato and others)
Valerio Formato, 22 months ago • Initial commit
master*
0 0
0
Live Share
Git Graph
Connected to Discord
UTF-8
LF
JSON
Prettier

```
{
  "type": "object",
  "properties": {
    "product": {
      "type": "object",
      "properties": {
        "title": {
          "type": "string",
          "minLength": 3,
          "description": "Title of your product. ex: TSST H-alpha Full Disk Images"
        },
        "shortname": {
          "type": "string",
          "minLength": 3,
          "description": "Product short name, for easier reference and ID build"
        },
        "type": {
          "type": "string",
          "enum": ["data", "lookup-table", "software"],
          "description": "Choose a type of the product: it affects the required metadata.'
        },
        "identifier": {
          "type": "string",
          "minLength": 1,
          "description": "Suggest a product unique identifier. Tentative form should be a:
        },
        "altidentifier": {
          "type": "array",
          "description": "optional, alternate product identifier(s), DOIs or other.",
          "items": {
            "type": "string"
          }
        },
        "created": {
          "type": "string",
          "format": "date-time",
          "description": "When this product description was first provided."
        },
        "updated": {
          "type": "string",
          "format": "date-time",
          "description": "Please update the datetime when product description changes."
        },
        "status": {
```

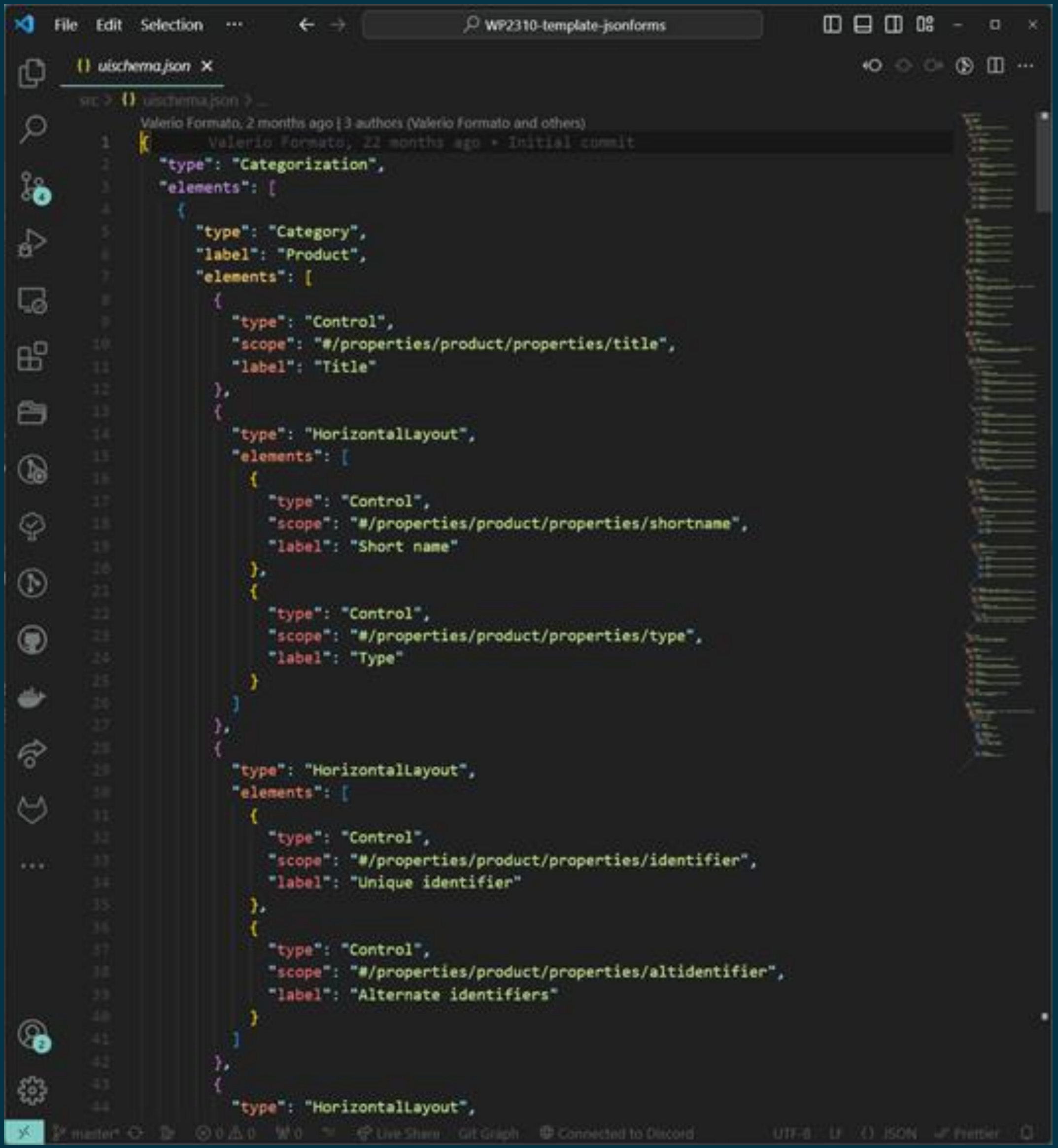
ProSpecT

The data in the forms will be stored in a JSON object.

JSONForms will use a special "schema.json" file to infer the final layout of the output JSON forms content.

Similarly, JSONForms requires a "uischema.json" file which specifies where and how the UI elements should be rendered in the resulting webpage.

UI elements are linked to a specific field in the JSON output via the special "scope" key.


File
Edit
Selection
WP2310-template-jsonforms
uischema.json
src > {} uischema.json > ...
Valerio Formato, 2 months ago | 3 authors (Valerio Formato and others)
master*
Live Share
Git Graph
Connected to Discord
UTF-8
LF
{} JSON
Prettier

```
{        Valerio Formato, 22 months ago • Initial commit
  "type": "Categorization",
  "elements": [
    {
      "type": "Category",
      "label": "Product",
      "elements": [
        {
          "type": "Control",
          "scope": "#/properties/product/properties/title",
          "label": "Title"
        },
        {
          "type": "HorizontalLayout",
          "elements": [
            {
              "type": "Control",
              "scope": "#/properties/product/properties/shortname",
              "label": "Short name"
            },
            {
              "type": "Control",
              "scope": "#/properties/product/properties/type",
              "label": "Type"
            }
          ]
        },
        {
          "type": "HorizontalLayout",
          "elements": [
            {
              "type": "Control",
              "scope": "#/properties/product/properties/identifier",
              "label": "Unique identifier"
            },
            {
              "type": "Control",
              "scope": "#/properties/product/properties/altidentifier",
              "label": "Alternate identifiers"
            }
          ]
        },
        {
          "type": "HorizontalLayout",
```

ProSpecT

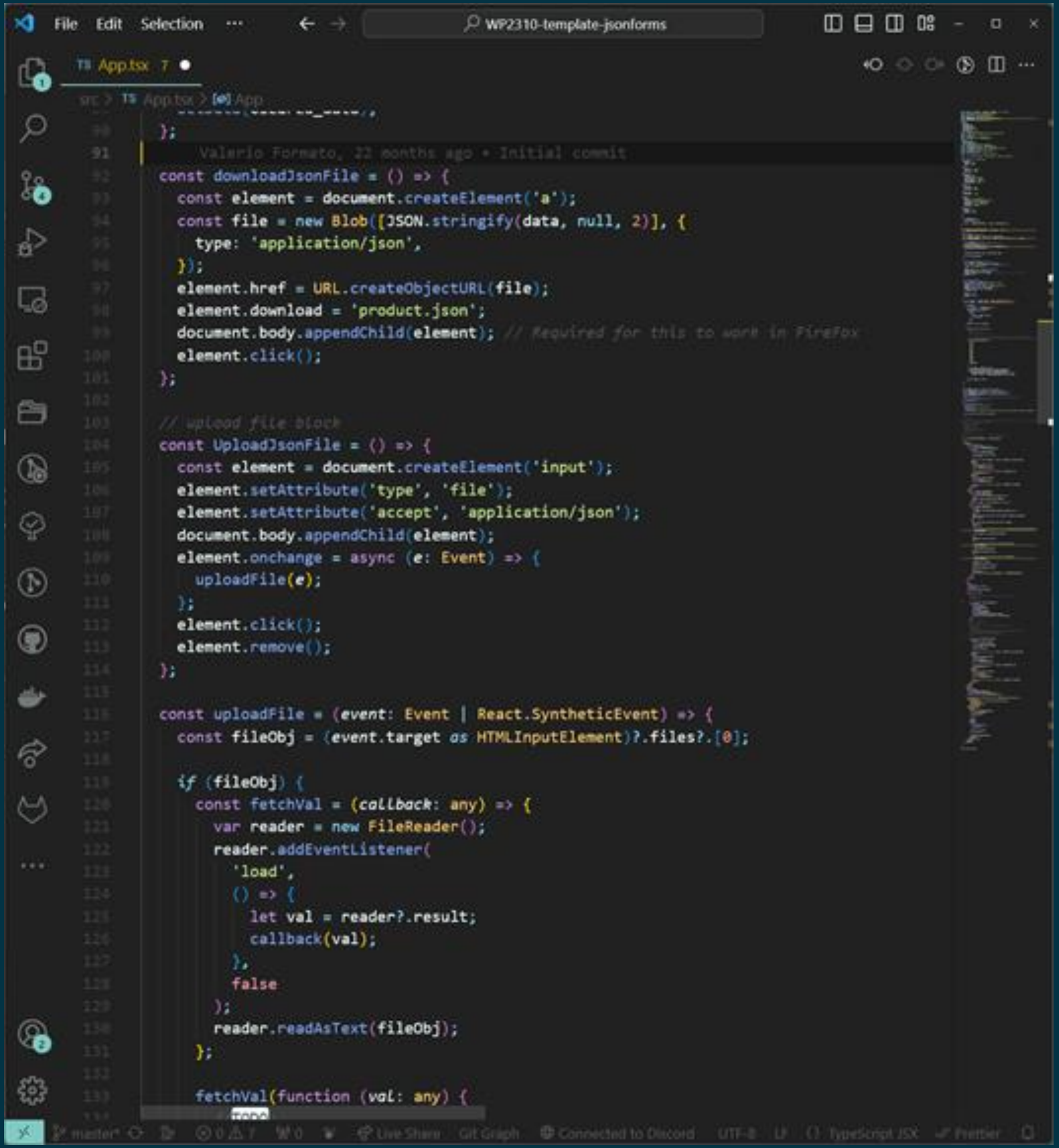
The data in the forms will be stored in a JSON object.

JSONForms will use a special “schema.json” file to infer the final layout of the output JSON forms content.

Similarly, JSONForms requires a “uischema.json” file which specifies where and how the UI elements should be rendered in the resulting webpage.

UI elements are linked to a specific field in the JSON output via the special “scope” key.

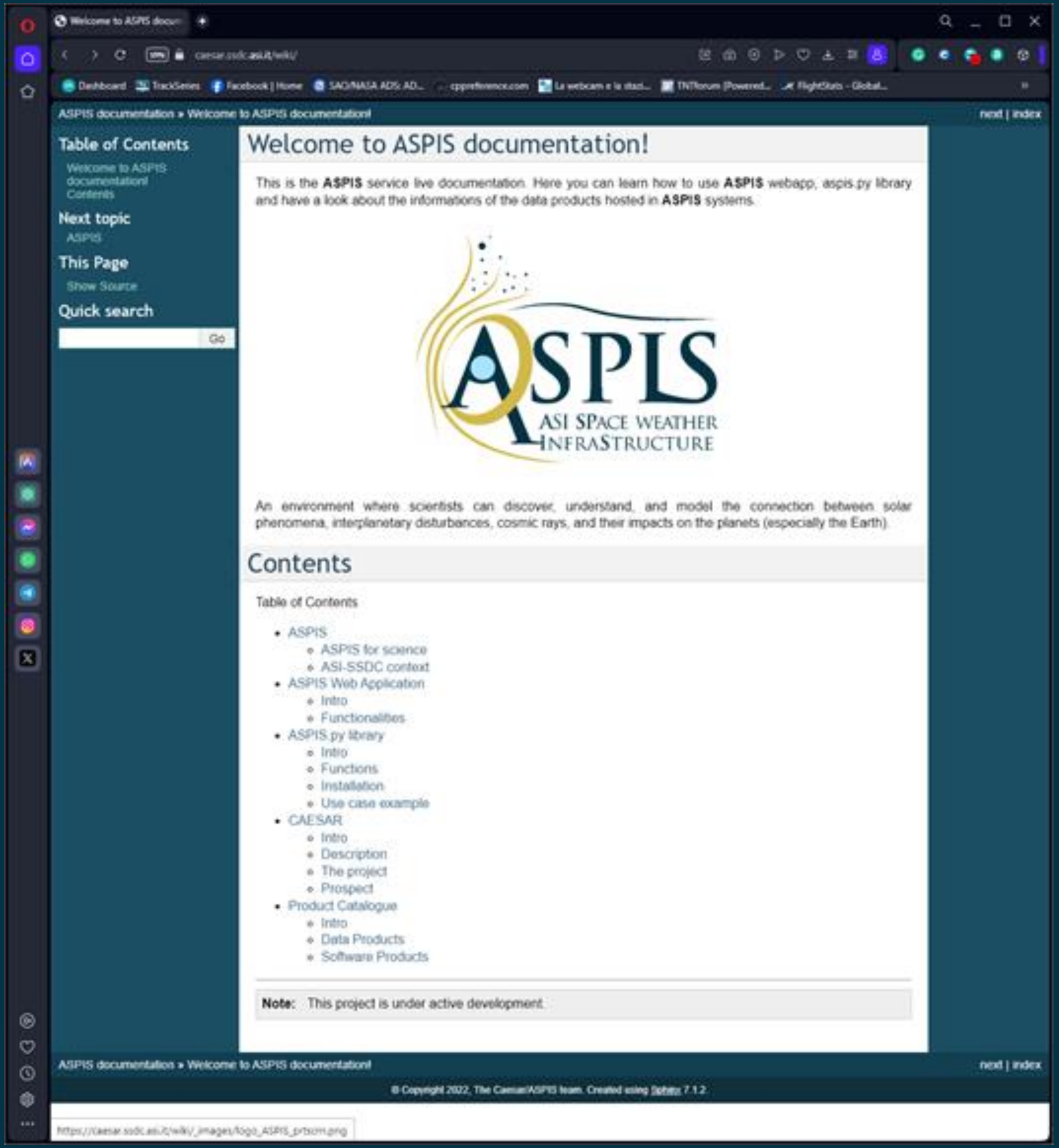
JSONForms supports different JS frameworks and custom features can be added to the main application.
e.g. ProSpecT users can save an incomplete form, and re-upload it later to finish working on it.



```
};

const downloadJsonFile = () => {
  const element = document.createElement('a');
  const file = new Blob([JSON.stringify(data, null, 2)], {
    type: 'application/json',
  });
  element.href = URL.createObjectURL(file);
  element.download = 'product.json';
  document.body.appendChild(element); // Required for this to work in FireFox
  element.click();
};

// upload file block
const UploadJsonFile = () => {
  const element = document.createElement('input');
  element.setAttribute('type', 'file');
  element.setAttribute('accept', 'application/json');
  document.body.appendChild(element);
  element.onchange = async (e: Event) => {
    uploadFile(e);
  };
  element.click();
  element.remove();
};

const uploadFile = (event: Event | React.SyntheticEvent) => {
  const fileObj = (event.target as HTMLInputElement)?.files?.[0];

  if (fileObj) {
    const fetchVal = (callback: any) => {
      var reader = new FileReader();
      reader.addEventListener(
        'load',
        () => {
          let val = reader?.result;
          callback(val);
        },
        false
      );
      reader.readAsText(fileObj);
    };

    fetchVal(function (val: any) {
```

ProSpecT

One of the results of this effort can be see in the accompanying wiki, which also holds a list of all the products included in the ASPIS prototype database



Welcome to ASPIS documentation!

This is the **ASPIS** service live documentation. Here you can learn how to use **ASPIS** webapp, aspis.py library and have a look about the informations of the data products hosted in **ASPIS** systems.

An environment where scientists can discover, understand, and model the connection between solar phenomena, interplanetary disturbances, cosmic rays, and their impacts on the planets (especially the Earth).

Contents

Table of Contents

- ASPIS
  - ASPIS for science
  - ASI-SSDC context
- ASPIS Web Application
  - Intro
  - Functionalities
- ASPIS.py library
  - Intro
  - Functions
  - Installation
  - Use case example
- CAESAR
  - Intro
  - Description
  - The project
  - Prospect
- Product Catalogue
  - Intro
  - Data Products
  - Software Products

Note: This project is under active development.

© Copyright 2022, The Caesar/ASPIS team. Created using Sphinx 7.1.2.

https://caesar.ssdc.asi.it/wiki/_images/logo_ASPIS_prtscrn.png

ProSpecT

One of the results of this effort can be see in the accompanying wiki, which also holds a list of all the products included in the ASPIS prototype database

The full list is generated from the collection of JSON metadata files that data providers in Node 1000 created using ProSpecT

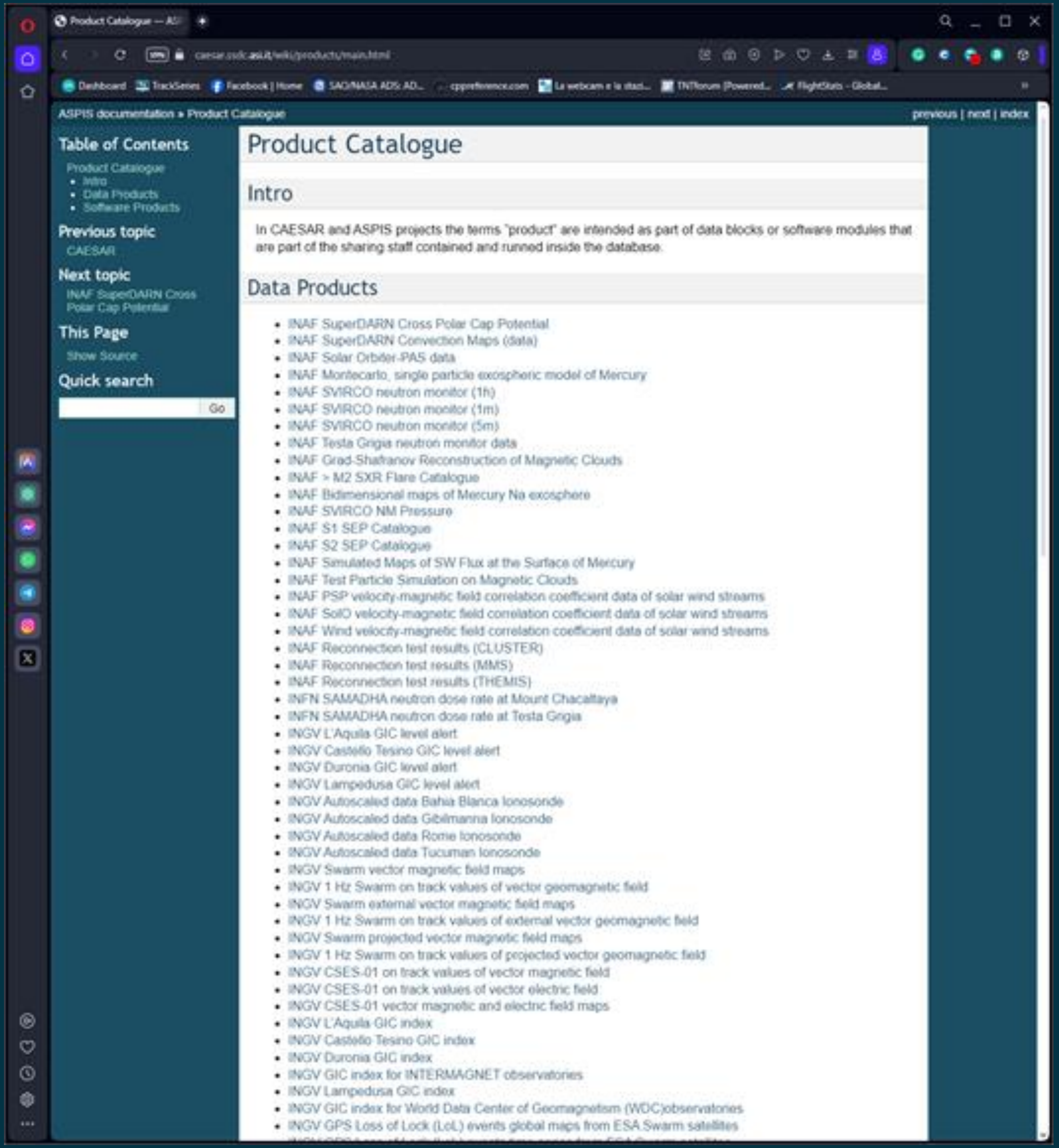


Table of Contents

Product Catalogue

- Intro
- Data Products
- Software Products

Previous topic

CAESAR

Next topic

INAF SuperDARN Cross Polar Cap Potential

This Page

Show Source

Quick search

Go

Product Catalogue

Intro

In CAESAR and ASPIS projects the terms "product" are intended as part of data blocks or software modules that are part of the sharing staff contained and runned inside the database.

Data Products

- INAF SuperDARN Cross Polar Cap Potential
- INAF SuperDARN Convection Maps (data)
- INAF Solar Orbiter-PAS data
- INAF Montecarlo, single particle exospheric model of Mercury
- INAF SVIRCO neutron monitor (1h)
- INAF SVIRCO neutron monitor (1m)
- INAF SVIRCO neutron monitor (5m)
- INAF Testa Grigia neutron monitor data
- INAF Grad-Shafranov Reconstruction of Magnetic Clouds
- INAF > M2 SXR Flare Catalogue
- INAF Bidimensional maps of Mercury Na exosphere
- INAF SVIRCO NM Pressure
- INAF S1 SEP Catalogue
- INAF S2 SEP Catalogue
- INAF Simulated Maps of SW Flux at the Surface of Mercury
- INAF Test Particle Simulation on Magnetic Clouds
- INAF PSP velocity-magnetic field correlation coefficient data of solar wind streams
- INAF SolO velocity-magnetic field correlation coefficient data of solar wind streams
- INAF Wind velocity-magnetic field correlation coefficient data of solar wind streams
- INAF Reconnection test results (CLUSTER)
- INAF Reconnection test results (MMS)
- INAF Reconnection test results (THEMIS)
- INFN SAMADHA neutron dose rate at Mount Chacaltaya
- INFN SAMADHA neutron dose rate at Testa Grigia
- INGV L'Aquila GIC level alert
- INGV Castello Tesino GIC level alert
- INGV Duronia GIC level alert
- INGV Lampedusa GIC level alert
- INGV Autoscaled data Bahia Blanca Ionosonde
- INGV Autoscaled data Gibilmanna Ionosonde
- INGV Autoscaled data Rome Ionosonde
- INGV Autoscaled data Tucuman Ionosonde
- INGV Swarm vector magnetic field maps
- INGV 1 Hz Swarm on track values of vector geomagnetic field
- INGV Swarm external vector magnetic field maps
- INGV 1 Hz Swarm on track values of external vector geomagnetic field
- INGV Swarm projected vector magnetic field maps
- INGV 1 Hz Swarm on track values of projected vector geomagnetic field
- INGV CSES-01 on track values of vector magnetic field
- INGV CSES-01 on track values of vector electric field
- INGV CSES-01 vector magnetic and electric field maps
- INGV L'Aquila GIC index
- INGV Castello Tesino GIC index
- INGV Duronia GIC index
- INGV GIC index for INTERMAGNET observatories
- INGV Lampedusa GIC index
- INGV GIC index for World Data Center of Geomagnetism (WDC)observatories
- INGV GPS Loss of Lock (LoL) events global maps from ESA Swarm satellites

Conclusions

The CAESAR project was created with the specific goal of implementing a prototype for the ASPIS project.

A large number of science cases and data products will be handled by the project. To facilitate the design process, and in future help querying for data products in ASPIS, a robust set of metadata should accompany each product.

JSON was chosen as the data format for metadata description.
To facilitate creation of such metadata files a dedicated tool (ProSpecT) was created using the JSONForms framework.

The resulting metadata files have been successfully used to automate large tasks (such as the creation of documentation) and are still largely used in the final design of the ASPIS prototype database.

Accordo ASI - INAF n. 2020-35-HH.O

Agenzia Spaziale Italiana

INAF ISTITUTO NAZIONALE DI ASTROFISICA NATIONAL INSTITUTE FOR ASTROPHYSICS

CÆSAR
COMPREHENSIVE SPACE WEATHER STUDIES FOR THE ASPIS PROTOTYPE REALIZATION

23/11/2023 ESWW 2023

Thank you!

CAESAR NODE 2000 Team
Valerio Formato (INFN), Marco Molinaro (INAF), Valeria di Felice (INFN), Dario Del Moro (UNITOV), Stefano Scardigli (UNITOV), Cristina Campi (UNIGE), Federico Benvenuto (UNIGE), Carmelo Magnafico (INAF), Rossana De Marco (INAF), Ermanno Pietropaolo (UNIAQ), Gregoire Francisco (UNITOV), Andrea Tacchino (UNIGE), Mirko Stumpo (INAF), Liu Scigè John (INAF), Giuseppe Di persio (INAF), Emanuele Scalise (INAF), Raffaella Noschese (INAF), Monica Laurenza (INAF)

EB Team
Cristina Plainaki, Anna Milillo, Giuseppe Sindoni, Marco Giardino, Alberto Bigazzi, Gianluca Polenta

Project Prime:

INAF ISTITUTO NAZIONALE DI ASTROFISICA NATIONAL INSTITUTE FOR ASTROPHYSICS

Project Partners:

INFN Istituto Nazionale di Fisica Nucleare

ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA

TOR VERGATA UNIVERSITÀ DEGLI STUDI DI ROMA

unipg UNIVERSITÀ DEGLI STUDI DI PERUGIA

Università di Genova

UNIVERSITÀ DEGLI STUDI DELL'AQUILA

UNIVERSITÀ DELLA CALABRIA

Università di Catania

UNIVERSITÀ DI TRENTO