JOHNS HOPKINS
APPLIED PHYSICS LABORATORY

HAPI

Using HAPI as a timeseries data access standard in an IVOA Context

interoperable access for time series data

Jon Vandegriff, JHU / APL
Baptiste Cecconi, Observatoire de Paris

Defining what is meant by time series data

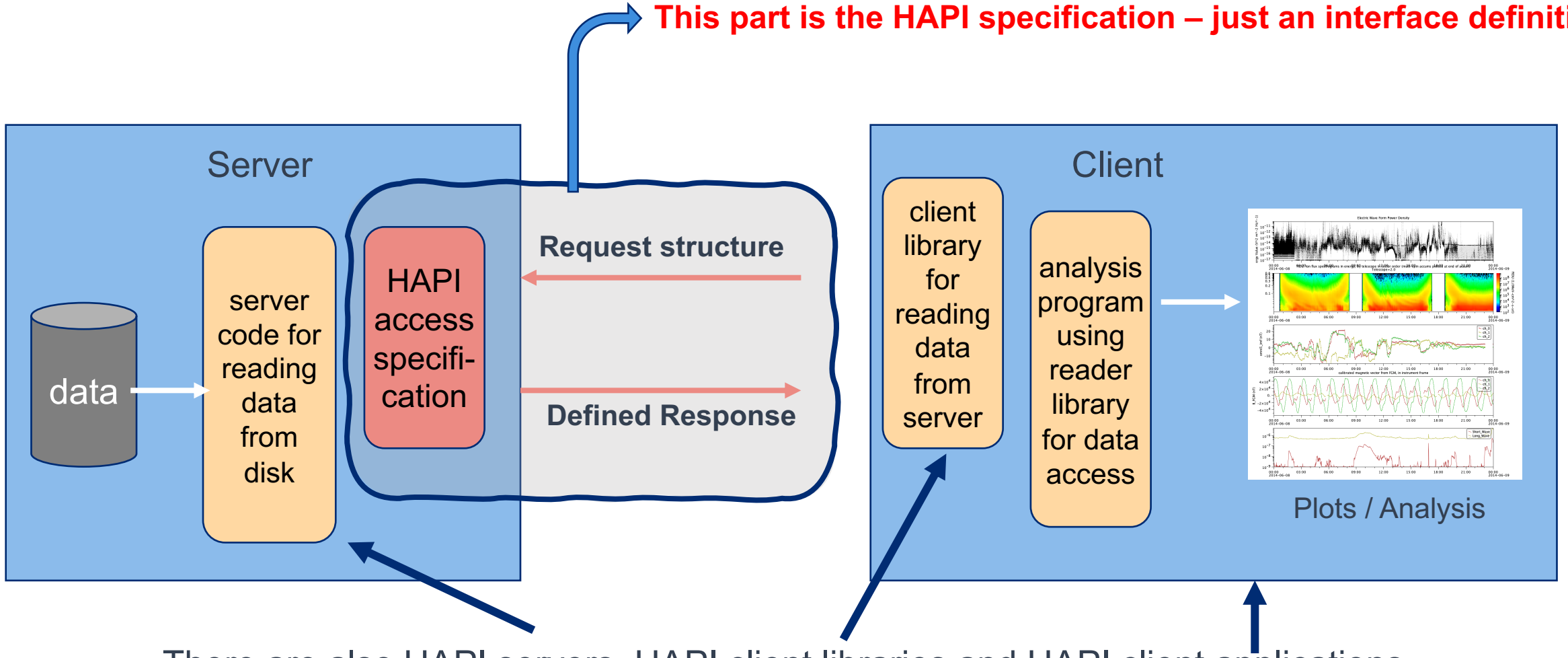
Time (as ISO 8601)	data1	scalar2	1-D array	multi-Dim Array
2023-01-10T14:20:12.456	d0	s0	a0[11]	m0[3,8]
2023-01-10T15:20:12.456	d1	s1	a1[11]	m1[3,8]
2023-01-10T16:20:12.456	d2	s2	a2[11]	m2[3,8]
2023-01-10T17:20:12.456	d3	s3	a3[11]	m3[3,8]
2023-01-10T18:20:12.456	d4	s4	a4[11]	m4[3,8]
2023-01-10T19:20:12.456	d5	s5	a5[11]	m5[3,8]
2023-01-10T12:20:12.456	d6	s6	a6[11]	m6[3,8]

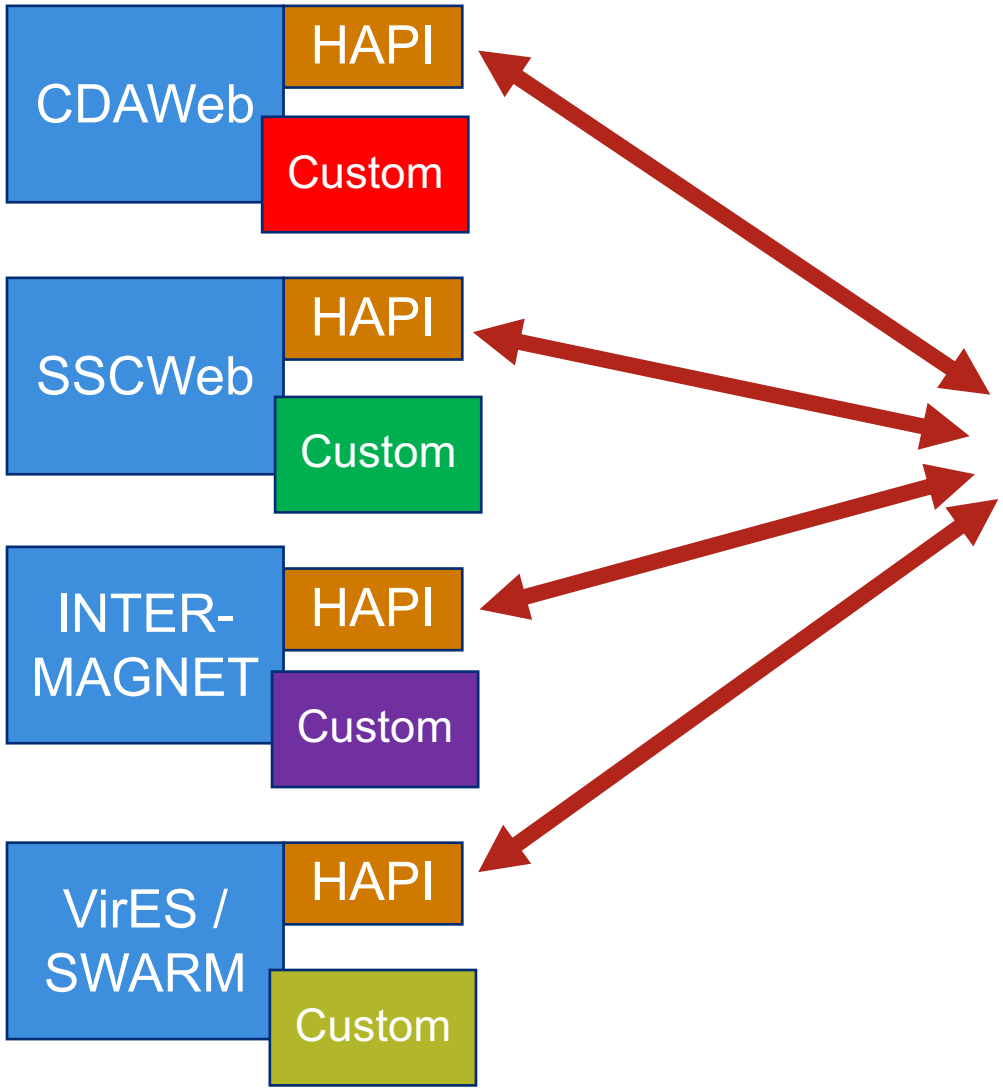
- conceptually, it is a table, like a spreadsheet
- time column followed by data columns (variables)
- each variable can be multidimensional (i.e., spectra, or data cubes in each cell)

… and keeps going at this cadence for **years or decades**

What is HAPI?

- HAPI = Heliophysics Application Programmer's Interface
- **contains no Heliophysics-specific features**





The point of HAPI is interoperability of data.

Have a data provider **add** a HAPI capability so that content from each can be reached with a single API. (keep original custom interface!)

These are some existing Heliophysics data services for space-based or ground-based observatories.

HAPI Works because it is very simple

- Think of HAPI as "**http for data**" – a protocol for asking and getting something from a server
- RESTful – no state so that each request is independent
- URLs representing the requests can be thought of as (semi-)persistent identifiers
- Endpoints define the things you can ask of a HAPI server

elements required by the data endpoint protocol

data endpoint

Request:

```
http://server.org/hapi/data?dataset=ACE_MAG
                           &start=2004-183T00:00Z
                           &stop=2004-184T00:00Z
```

(line breaks for clarity)

Response:

```
2004-183T00:00:03.403Z,  1.0724e+02, -6.8993e+01, -5.1978e+02
2004-183T00:00:07.153Z,  1.0842e+02, -6.8956e+01, -5.1962e+02
2004-183T00:00:10.907Z,  1.0855e+02, -6.9063e+01, -5.2084e+02
2004-183T00:00:14.653Z,  1.0852e+02, -6.9049e+01, -5.2085e+02
2004-183T00:00:18.403Z,  1.0849e+02, -6.9035e+01, -5.2085e+02
2004-183T00:00:22.153Z,  1.0862e+02, -6.9142e+01, -5.2207e+02
2004-183T00:00:25.903Z,  1.0859e+02, -6.9128e+01, -5.2208e+02
```

(always the same format for all servers)

HAPI defines 5 URL endpoints every server must have

Endpoints must be directly below a URL that ends with '**`hapi`**'

- `http://example.com/hapi/about`
- `http://example.com/hapi/capabilities`
  - describes options implemented by the server
- `http://example.com/hapi/catalog`
  - list of datasets at the server
- `http://example.com/hapi/info`
  - show metadata for one dataset at a time (basically a data header)
- `http://example.com/hapi/data`
  - retrieve a stream of data content for one dataset over a specific time range

Note: The intent is for **computers** to read from these endpoints, but humans can look at them easily too (web browser, curl, etc)

HAPI `info/` response example (JSON metadata)

```
{
  "HAPI": "2.0",
  "status": {"code": 1200, "message": "OK"},
  ”startDate": "1997-09-02T00:00:00Z",
  "stopDate": "2023-08-20T23:00:00Z",
  "resourceURL": "https://cdaweb.gsfc.nasa.gov/misc/NotesA.html#AC_H2_MFI",
 "contact": "N. Ness @ Bartol Research Institute”   "parameters": [
    { "name": "Time", "type": "isotime", "units": "UTC", "length":24, "fill": null },
    { "name": "Magnitude", "type": "double", "units": "nT", "fill": "-1.0E31",
            "description": "B-field magnitude" },
   { "name": "BGSEc", "type": "double", "units": "nT", "fill": "-1.0E31", "size": [3],
            "description": "Magnetic Field Vector in GSE Cartesian coordinates (1 hr)"},
   { "name": "BGSM", "type": "double", "units": "nT", "fill": "-1.0E31", "size": [3],
            "description": "Magnetic field vector in GSM coordinates (1 hr)", },
   { "name": "SC_pos_GSE", "type": "double", "units": "km", "fill": "-1.0E31", "size": [3],
            "description": "ACE s/c position, 3 comp. in GSE coord."},
   { "name": "SC_pos_GSM", "type": "double", "units": "km", "fill": "-1.0E31", , "size": [3],
            "description": "ACE s/c position, 3 comp. in GSM coord.” } ],
}
```

Python code stub generated for you

https://hapi-server.org/servers *(an exploratory HAPI client as JavaScript in a web page that has a service that can generate code examples)*

```
# example showing how to get OMNIWeb data
from hapiclient import hapi

server     = 'https://cdaweb.gsfc.nasa.gov/hapi'
dataset    = 'OMNI2_H0_MRG1HR'
start      = '2021-10-25T00:00:00Z'
stop       = '2021-12-01T00:00:00Z'
# parameters is a comma-separated list
parameters = 'DST1800,Proton_QI1800'

# Configuration options for the hapi function.
opts = {'logging': True, 'usecache': True, 'cachedir': './hapicache' }

# Get parameter data
data, meta = hapi(server, dataset, parameters, start, stop, **opts)
```

easy to change the date, for example (shown in red)

Benefits of Using HAPI

- Simplicity – easy to use data that is exposed using HAPI:
  - all file storage details (files, databases, etc) are hidden => data response is a stream of rows
  - there are existing Python and Java client libraries people can use for access
  - we have a service that can generate code stubs for access (see next slide)
- Simplicity again on the data provider side:
  - very similar to what all providers are already using
    - almost all implementers have opted to just modify existing servers to add HAPI
    - (we have generic servers, but HAPI is so close to what people use already, it is easy to adapt)
  - metadata requirements are minimal – just what's needed to plot the data
- HAPI will also work cloud-to-cloud
  - working on this now, but don't' have it in place yet

HAPI and the IVOA

- HAPI came out of different community
- offers: single API for accessing lot of Heliophysics and Planetary data!
- could be leveraged in astronomy for time series (light curves)
  - which provider could be interested?

- How to integrate into IVOA ecosystem?
  - **It must be kept simple to implement** (this as the key for success and adoption).
  - Explore use of UCD in header and info?
  - Explore implementation of provenance? (for traceability of progenitor data)
  - Registry: check how to declare HAPI service?
  - Output: streamed data in the VO => VOTable (RemoteData) + Binary stream HAPI URL ?
  - Implement a SODA layer on top of HAPI to translate the queries?