AI agents can now query VESPA

properly — not freewheeling

S. Erard — Astro-CC forum 2 · October 2026

Europlanet / VESPA

[Triggered by Carlo's presentation on VAMDC]

And, yes, presentation partly generated by AI – under constraints, but it's a bit pretentious

The problem

Colleagues try AI assistants on their data. Too often the agent:

- invents a plausible-looking answer
- mixes incompatible quantities
- hammers servers with silly queries

Not because AI is bad at science — because it was never told how our services work

The fix: a guide the agent reads

`query-epntap` — a plain Markdown file, next to our tutorials:

- where the services are, and how to talk to them (TAP/ADQL)
- the traps, e.g., SI units (JD, Hz), schema names, measurement scales
- the rules: cite the granule, don’t crawl, say when the answer is partial

Documentation similar to our tutorials — written for a machine reader

- Also tells how to access the local registry of the portal, and the portal itself

What it looks like

You say:

"Show me a near-infrared spectrum of Ceres"

The agent: finds the right service, converts 0.7–2.5 µm to Hz, picks reflectance — not flux — plots the real data file, cites the paper.

Seconds instead of reading a tutorial. The user asks in plain language.

Very efficient to reach new / potential users

Every claim is labelled

Each statement in the skill carries its evidence:

- **tested** — executed against the servers, dated
- **expert** — stated by a domain expert
- **written** — unverified, flagged as such

If a rule is in there, you know where it came from.

Good practices imposed

Fixes after initial issues:

- Follows authoring guidelines from **Skills Commons** (https://skills-commons.org/write/)

- Applies research-integrity rules from **Lu et al. 2026**, Nature 651, 914 (doi:10.1038/s41586-026-10265-5)
  — operationalized in https://github.com/Imbad0202/academic-research-skills (academic-pipeline/references/ai_research_failure_modes.md)

Safety rails, built in

- **No fabrication**: figures are plotted from the data file, never improvised — with a mechanical check (point count, axis bounds). This rule exists because we caught one real case.
- **No server/portal abuse**: 1–2 discovery calls, then direct TAP; no loops, no crawling; agents identify themselves (`tapclient`) so **we can see the traffic**
- **No silent automation**: scheduled/bulk use is out of contract — this is for real users asking real questions

End to end test

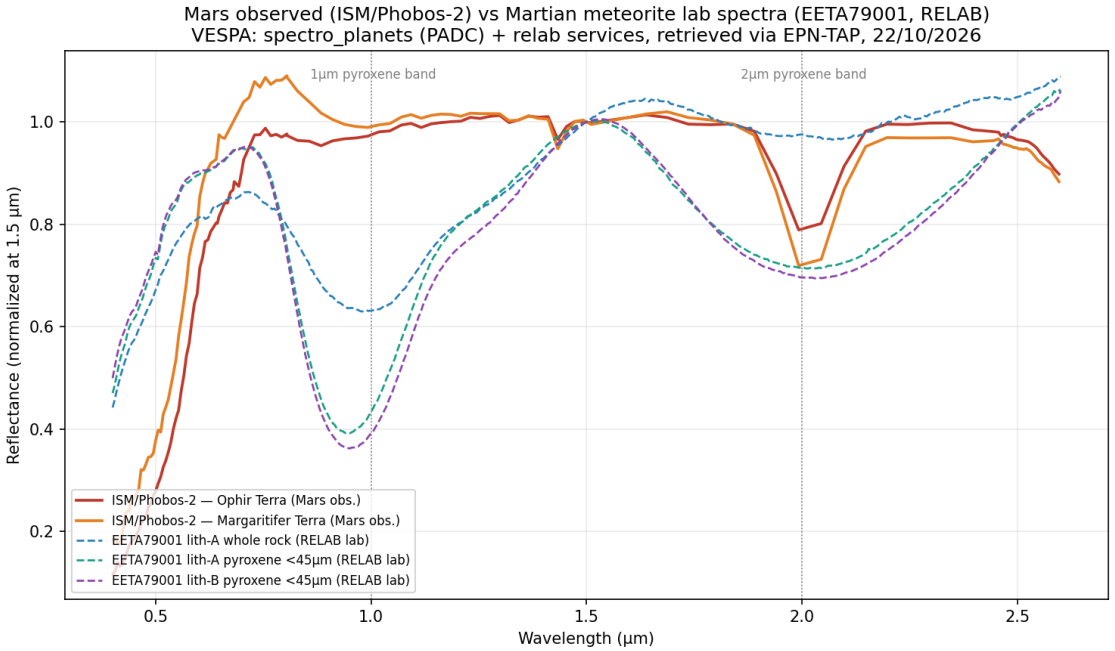
Prompt: compare spectra of Martian meteorites with Martian spectra

- Finds meteorites spectra in RELAB, identifies SNCs
- Finds OMEGA_cubes at iDOC, identifies region of interest – but disk-storage is not reachable
- => Backup plan: looks for Mars spectra in other services
- 5 VOTable, manages to read them – with an ad hoc parser (no astropy)
- Plots spectra, decides cutout & normalization
- Then, a bit enthusiastic with interpretation:
  - Publishes band centers, but only from location of minimum - very approximative
  - Pushes interpretation too far (sees pyroxenes in bright regions, this is FeO)
  - Misinterprets obvious artefacts (atmospheric remnants)

End to end test

Q: compare spectra of Martian meteorites with Martian spectra

Altogether rather impressive: answers a not-so-basic question by grabbing data in several services, uses a backup plan on failures, makes reasonable choices — **Convincing VO demonstrator** —



mars_meteorite_comparison.py

```python
import xml.etree.ElementTree as ET
import numpy as np, matplotlib
matplotlib.use('Agg')
import matplotlib.pyplot as plt

def parse_vot(path):
    tree = ET.parse(path); root = tree.getroot()
    ns = root.tag.split('}')[0].strip('{')  # detect namespace
    T = lambda tag: f'{{{ns}}}{tag}'
    tbl = root.find(f'.//{T("TABLE")}')
    fields = tbl.findall(f'.//{T("FIELD")}')
    names = [f.get('name').lower() for f in fields]
    rows = tbl.findall(f'.//{T("TR")}')
    data = []
    for tr in rows:
        vals = [float(td.text) for td in tr.findall(T('TD'))]
        data.append(vals)
    arr = np.array(data)
    return names, arr

specs = {}
for key, path, lab in [
    ('ism_ophir',      '/tmp/ism_ophir.vot',      'ISM/Phobos-2 — Ophir Terra (Mars obs
    ('ism_margarit',   '/tmp/ism_margaritifer.vot','ISM/Phobos-2 — Margaritifer Terra (
    ('eeta_lithA_rock','/tmp/relab_C1DD19.vot',   'EETA79001 lith-A whole rock (RELAB l
    ('eeta_lithA_pyr', '/tmp/relab_C1DD57.vot',   'EETA79001 lith-A pyroxene <45µm (REL
    ('eeta_lithB_pyr', '/tmp/relab_C1DD58.vot',   'EETA79001 lith-B pyroxene <45µm (REL
```

Returns Python code to read & plot the files

Human comment: AIs are pranksters

- Skills are ~ always accepted as text indeed

But you need an environment with network access – won't work otherwise

I have tested it with Mistral/Vibe (I happen to have a pro account), not anywhere else

=> Degraded mode implemented: In case of impossibility, the agent will provide an SQL request or python code instead of a result, already a first step for new users

- Need to pass agent ID to data servers, also through the VESPA portal
- May be a good idea to split this in pieces, with skills for pyvo, TAP, etc (TBC)

Try it, break it, tell us

Install the skill, ask for your data.
Every failure you report becomes a rule — and often a fix.
Available in the VESPA tutorials repo: *https://github.com/epn-vespa/tutorials*
CC-BY 4.0 — assessment phase

AI agent skills exist: one TAP/pyvo/ADQL on https://github.com/skill-commons/skill-commons
+ some astropy + other fields, including VAMDC
This is apparently the first agent skill for Solar System VO access

— inspired by Carlo's VAMDC presentation, developed in 24h