Topical Discussion Meeting 03 Report

“THE SPACE ENVIRONMENT INFORMAYION SYSTEM – A NEW FRAMEWORK”

Conveners: Erwin de Donder (BIRA-IASB), Neophytos Messios (BIRA-IASB)
Secretary: Daniel Heynderickx (DH Consultancy)
Contact email: erwin.dedonder@aeronomie.be, spenvis_team@aeronomie.be
Location: Cassiopée Room

1. Introduction (presentation by E. De Donder)

See slides in annex.

2. Network of Models (presentation by S. Clucas)

See slides in annex.

3. Live demonstration (performed by S. Mezhoud)

4. Discussion

Q: Is it possible to compute the GCR (or SEP) for any position in the solar system, and for any specified time?
A: Currently, GCR fluxes are calculated at 1 AU. Magnetic and planetary shielding will be added later (also for SEP flux).

Q: Is the mission tool already implemented?
A: No, not yet.

Q: Is it possible to specify a physics list for Geant4 applications or does the system handle this?
A: There is an option for automatic physics list generation, or the user can define their own lists. NoM also allows to upload a macro file, so the user has full control.

Q: Are there any tools available to generate an EOR trajectory?
A: OHB mission designers provide a text file with the orbit. I can find out which tools they are using. Also, check about file formats.

Q: SPENVIS-4 uses the ECSS rule to distance scale SEP fluxes. I want to be able to override this.
A: There will be a separate distance scaling application which will allow scaling laws other than ECSS.

Q: Is there a requirement for model providers to use dockerized implementations?
A: NoM uses the Geant4 dockerized implementation given the complexity of installation of the codes. Model providers can also use dockerized implementations, or provide a Fortran binary, for instance. Model providers do need to provide sufficient documentation.

Q: Is it possible to implement NUMIT in SPENVIS? What is the process?
A: In principle, yes, if ESA agrees. Supplying a model binary and documentation should be sufficient.

Q: A nice feature for the wish list: automatic report generation in PDF or Word format.

Q: How difficult or easy is it to install and run a NoM server?
A: It is straightforward, the NoM server is a Flask application.

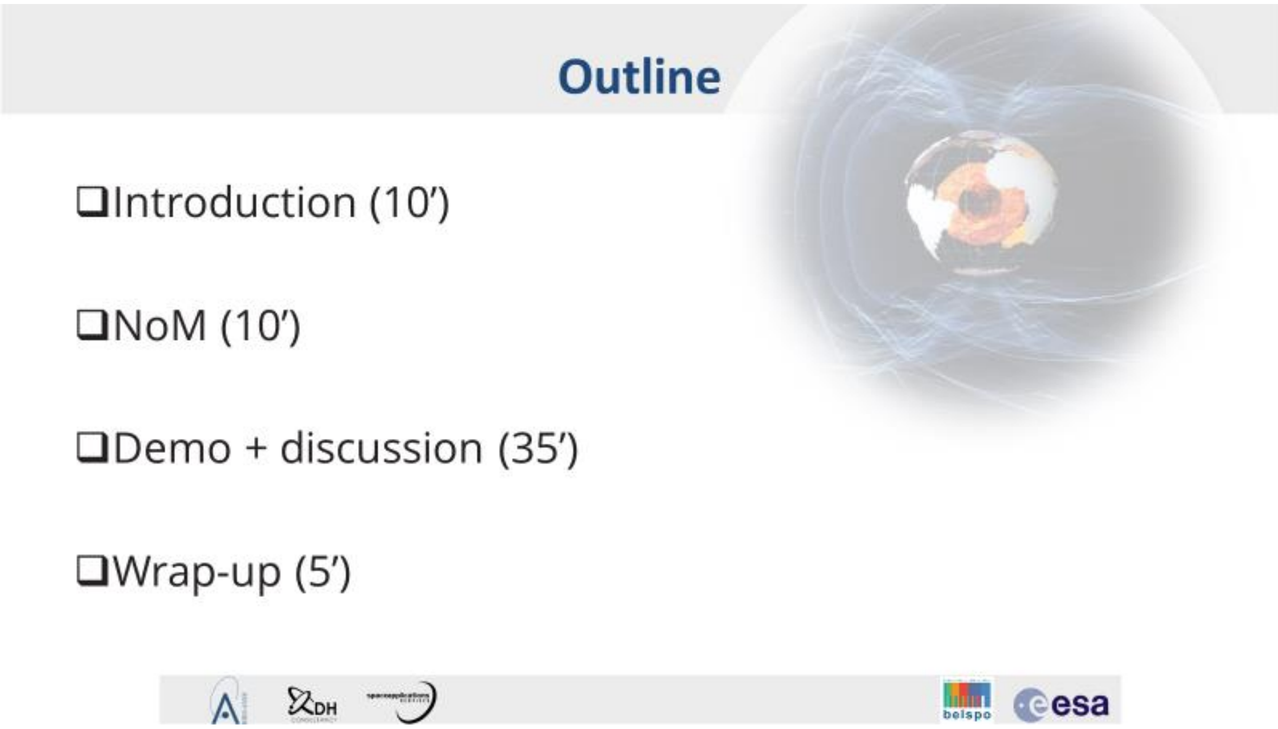
Q: Do I need to administer server keys? Can keys be used in NoM chain calls?
A: At this point, key administration is very simple, there is no admin tool.

5. Abbreviations

AU	Astronomical Unit
BIRA-IASB	Royal Belgian Institute for Space Aeronomy
ECSS	European Cooperation for Space Standardization
EOR	Electric Orbit Raising
ESA	European Space Agency
GCR	Galactic Cosmic Rays
Geant	GEometry ANd Tracking
NoM	Network of Models
NUMIT	NUMerical InTegration (Internal charging code)
OHB	Otto Hydraulik Bremen
SEP	Solar Energetic Particles
SPENVIS	Space ENVironment Information System

6. Annex

A.1. SPENVIS introduction



Motivation

SPENVIS-4 limitations:

- ❑ **Model-executed driven system**
  - rigid workflow for model access + no flexibility in model coupling
- ❑ **Lack of flexibility**
  - start: mission definition (>set of segments>set of orbits → trajectory → env. model)

  → not possible to select and specify an environment model per mission segment
- ❑ **Organic structure**
  - difficult to integrate new model
- ❑ **Lack of granularity**
  - compound codes → complexity in model interface and output
- ❑ **Lack of API, interoperability with other software tools**
- ❑ ...

New SPENVIS – "SPENVIS-5"

❑ **Context:**

ESA General Support Technology Programme (be)
G617-248EE: SPENVIS(-NG) interfaces, tools and models
4000134504/21/NL/CRS

❑ **ESA Technical officer:**

Simon Clucas (ESTEC/TEC-EPS)

❑ **Consortium:**

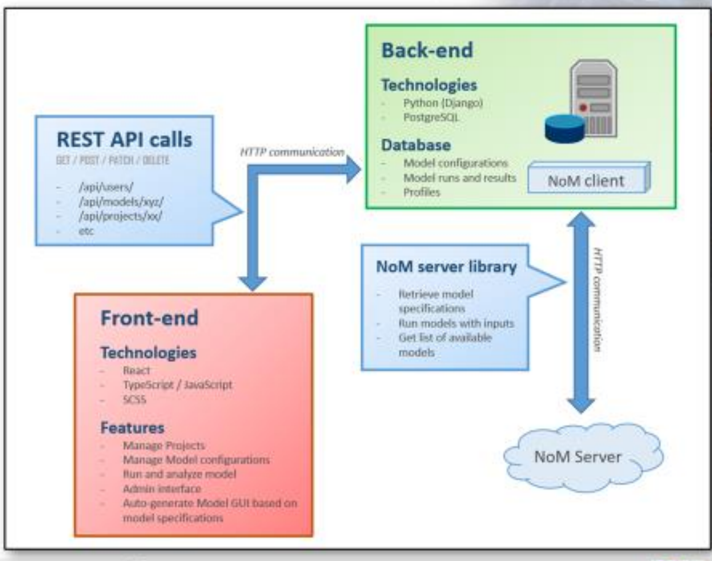
BIRA-IASB, DH Consultancy bv, Space Applications Services nv

Main objectives

- ❑ Refactor existing SPENVIS-4 models to allow more flexibility in the way they are used and combined
- ❑ Redesign SPENVIS front-end to improve the user experience and provide a consistent and expandable interface
- ❑ Implementation of API
- ❑ Integration of new models

→ New system from scratch

Framework architecture



New trajectory tool

➥ *Creation of individual trajectory segments in individual files.*

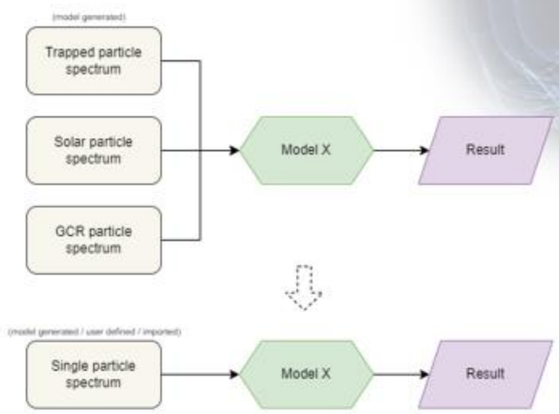
- SAPRE routines for trajectories around Earth
- NAIF/SPICE toolkit for Keplerian orbits around other planets and major moons in solar system
- Trajectory specification:
    - User inputs (classical ephemera, state vector, etc.)
    - Uploaded spacecraft coordinates/vectors

    → Support TLE, LTOF, OEM files
- Output full state vectors in Cartesian J2000 reference frame

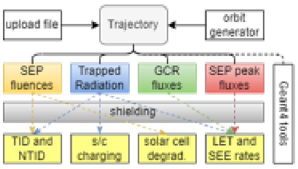
(https://bids.berkeley.edu/news/joy-code-refactoring)

Spectra consuming models

➥ Model should accept any (appropriate) single spectrum and create outputs based on that single input spectrum

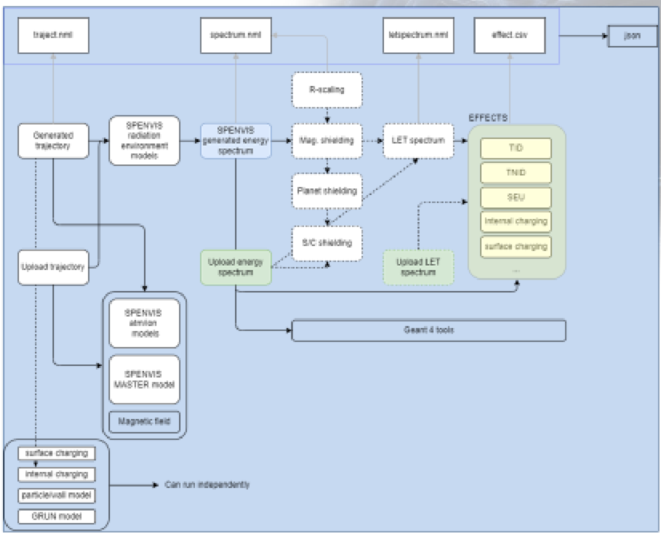


Workflows

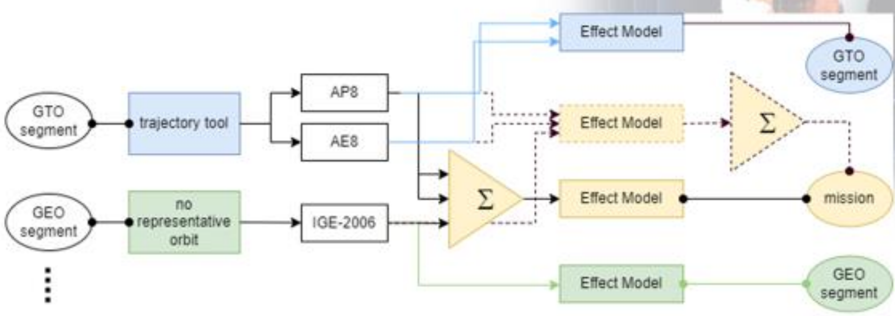


- ECSS workflows (LEO mission, MEO mission, …)
- User defined workflows

→ guidelines



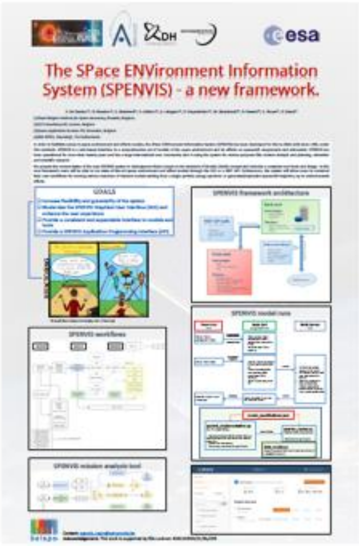
Mission analysis tool



New models

- Trajectory tool
- SPENVIS ODI → ACE, GOES, IREM, SREM, …
- IRI-**2016** (International Reference Ionosphere)
- DICTAT **4.1** (DERA Internal Charging Threat Assessment Tool)
- MCICT (Monte Carlo Internal Charging Tool, Lei et al., 2016)
- MASTER-**8** (ESA's Meteoroid and Space Debris Terrestrial Environment Reference Model)
- GRAS (Geant4 Radiation Analysis for Space) **5.0**
- DLR GCR (Matthia et al., 2013)
- BON2020 (Badhwar-O'Neill, 2020)
- LARB (Radiation Environment at Extremely Low Altitude and Latitude)
- GLORAB (Global radiation belt prototype for LEO constellations)
- …

SPENVIS poster - Session SWR04



SWR04-**1293**
Poster II, Wednesday 14:00 - Friday 12:00
Thursday 10:15-11:45, Friday 10:15-11:45
Caravelle Poster Hall

Wrap-up

- Separate section on current SPENVIS homepage with info on new system
- Planning of a SPENVIS user workshop → transition to new system + 3rd party models
- UNILIB software package: https://essr.esa.int/project/unilib-magnetic-field-library
  - www.mag-unilib.eu

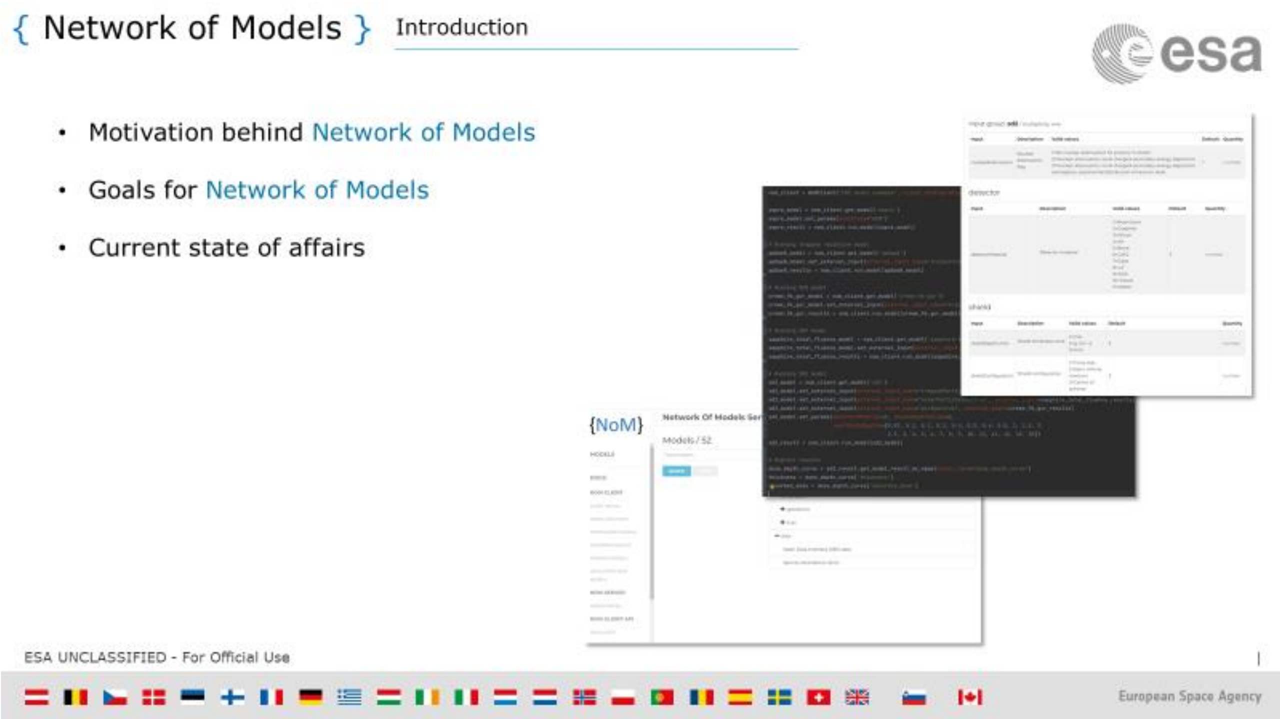
THANK YOU!

MORE INFO OR FEEDBACK?

spenvis_team@aeronomie.be

A.2. ESA NoM


esa
TDM 03 - THE SPACE ENVIRONMENT INFORMATION SYSTEM (SPENVIS) - A NEW FRAMEWORK
Network of Models Overview and Current State
Simon Clucas (ESA | ESTEC | Space Environments & Effects)
21/11/2023
ESA UNCLASSIFIED - For Official Use
European Space Agency


{ Network of Models } Introduction
esa
• Motivation behind Network of Models
• Goals for Network of Models
• Current state of affairs
{NoM}
ESA UNCLASSIFIED - For Official Use
European Space Agency

{ Network of Models } Motivation

A large number of models and data exist within the space environment and effects community

Written in…
- Fortran, C, C++, Python, Excel, VB …

Bundled as…
- complicated build systems with 20 year old dependencies
- huge VM images
- monolithic, closed source windows binaries
- elaborate web-based LAMP stack

Run using…
- web browser
- windows GUI point and click
- FORTRAN namelists
- GEANT4 macros

Giving results as …
- CSV files
- Binary files
- HTML
- Only stdout

{ Network of Models } Motivation

A large number of models and data exist within the space environment and effects community

Written in…
- Fortran, C, C++, Python, Excel, VB …

Bundled as…
- complicated build systems with 20 year old dependencies
- huge VM images
- monolithic, closed source windows binaries
- elaborate web based LAMP stack

Run using…
- web browser
- windows GUI point and click
- FORTRAN namelists
- GEANT4 macros

Giving results as …
- CSV files
- Binary files
- HTML
- Only stdout

- End users can spend significant time with IT and data issues and not their science and engineering
- These issues can be multiplied if you have a workflow that, for example, uses a tool to define the environment and several effects tools
- Often you need to iterate over a large parameter space which can be time consuming
- Models can exist with different implementations (SD2, ISO GCR …)
- Finally, getting the results from one tool into another tool may also require you to create another tool!

{ Network of Models } Goals

Single python client API to run all NoM models

- Easier discovery of models and data
- Less custom scripting and validating of data

Lightweight, general facade to convert any model into a **NoM server**

- Can retro-fit existing projects
- Simple interface spec for new projects
- More time on model development
- Less requirement to create custom API

Standardise on model description:
- Model docs
- Model meta data (version, provider etc.)
- Input provision format
- Output formats
- Dynamic model GUI creation

- Single-authoritative source of model information

Standardise on data types:
- Spectra, time series, data maps, images …
- Particle fluxes, LET, dose-depth …

- Model interoperability and pipelining
- Simplifies data analysis
- Simplifies plotting
- …

{ Network of Models } Current situation: NoM Server

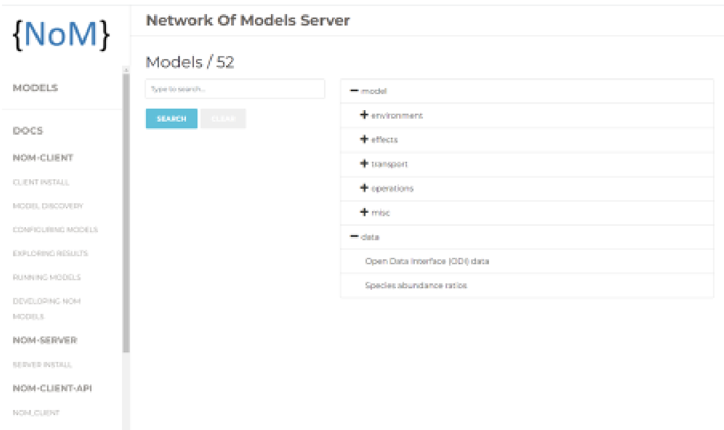
ESA NoM Server available:
- https://nom.esa.int

nom.esa.int has 52 models

- Searchable model table
- Model pages describe all inputs and outputs
- Model examples
- NoM Client documentation

Exposes Web API:
- Model discovery (nearly HAPI)
- Running models*
- Getting results*
- …

* Requires API key



All model documentation is generated from the actual model specifications

{ Network of Models } Current situation: Models

Types of models running on https://nom.esa.int

- Binary executables using input files (SD2, IRENE …)
- Binary executables + command line arguments (DLR GCR …)
- Python models (abundances …)
- Docker/containerised models (G4SpaceApps (GRAS, SSAT, MULASSIS))
- WebAPI/RestAPI/RPC models (ODI …)
- Composite, complex models using a combination of several models (SPLEEM, SEU Time Series Tool …)
- Models on other NoM Servers (soon)

Made possible because all models are wrapped by two simple interfaces:

- ModelImplementation
- OutputReader

Model providers need only create two python classes implementing the above interfaces to allow their model to be run through NoM

{ Network of Models } Current situation: NoM Client

- Allows model discovery based on keywords
- Converts the model specification into a python object which can be used to set inputs
- Runs models across several servers seamlessly
- Can poll long running jobs
- Provides methods to get and use results (slicing multi-dimensional data)
- Plugins can be written to provide result plotting or advanced analysis

```
# coding=utf-8

from nom_client.nom_client import NoMClient

conf = {
    "servers": {
        "ext_dev_rest_server": {
            "type": "rest",
            "server_addr": "https://nom.esa.int",
            "api_version": "api",
            "api_key": ""
        }
    }
}

nom_client = NoMClient("JOREM example", client_configuration=conf,
                       default_server_id="ext_dev_rest_server",
                       debug_output=False)

sapre_model_jupiter = nom_client.get_model('sapre')
sapre_model_jupiter.set_params(planet=5, apogeeAltitude=1811302, perigeeAltitude=181130)

sapre_jupiter_result = nom_client.run_model(sapre_model_jupiter)
print(sapre_jupiter_result)

dh_spicy_model = nom_client.get_model("greet")
dh_spicy_traj_res = nom_client.run_model(dh_spicy_model)
print(dh_spicy_traj_res)
```

Available at the ESA Space Environment & Effects Software repository
https://space-env-repo.estec.esa.int/network-of-models/code/nom-client.git

Access to this repository can be requested via: nom@esa.int

Running Shieldose2

Environment models accept a **trajectory** as external input

- sapre
- greet
- your model (needs to output a **trajectory**)

Using external data source ODI for SEP data

TID model accepts **particle spectra** as external inputs

- Ax8
- GCR
- SEP (SAPPHIRE etc.)
- External data (ODI ...)
- your model (needs to output a **particle spectra**

```
nom_client = NoMClient("SD2 model example", client_configuration=conf, default_server_id="nom_rest_server")

sapre_model = nom_client.get_model('sapre')
sapre_model.set_params(orbitType="GEN")
sapre_result = nom_client.run_model(sapre_model)

# Running trapped radiation model
ap8ae8_model = nom_client.get_model('ae8ap8')
ap8ae8_model.set_external_input(external_input_name="trajectory", external_input=sapre_result)
ap8ae8_results = nom_client.run_model(ap8ae8_model)

# Running GCR model
creme_96_gcr_model = nom_client.get_model('creme-96-gcr')
creme_96_gcr_model.set_external_input(external_input_name="trajectory", external_input=sapre_result)
creme_96_gcr_results = nom_client.run_model(creme_96_gcr_model)

# Get SEP spectrum from SEPEM RDS via ODI
odi_model = nom_client.get_model('odi-data')
odi_model.set_params(operation="read-data", dataSetName="sepem_rds_3_0", startDateTime="20031027T00:00:00", endDateTime="20031028T00:00:00")
sepem_rds_result = nom_client.run_model(odi_model)

# Running SD2 model
sd2_model = nom_client.get_model('sd2')
sd2_model.set_external_input(external_input_name="trappedParticleSpectrum", external_input=ap8ae8_results)
sd2_model.set_external_input(external_input_name="solarParticleSpectrum", external_input=sepem_rds_result)
sd2_model.set_external_input(external_input_name="gcrSpectrum", external_input=creme_96_gcr_results)
sd2_model.set_params(detectorMaterial=1, shieldDepthValues=1,
                     userShieldDepths=[0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.8, 1, 1.5, 2,
                                       2.5, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 20])
sd2_result = nom_client.run_model(sd2_model)

# Explore results
dose_depth_curve = sd2_result.get_model_result_by_name(result_name="dose_depth_curve")
thickness = dose_depth_curve['thickness'].values
absorbed_dose = dose_depth_curve['absorbed_dose'].values
```


Running Shieldose2

Environment models accept a **trajectory** as external input

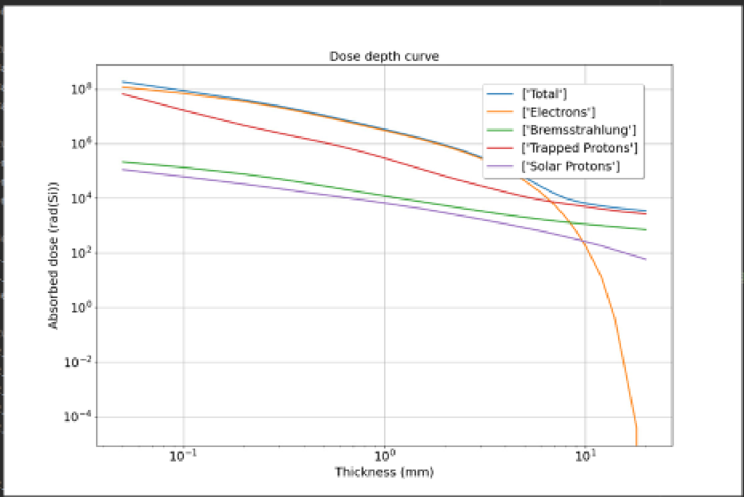
- sapre
- greet
- your model (needs to output a **trajectory**)

Using external data source ODI for SEP data

TID model accepts **particle spectra** as external inputs

- Ax8
- GCR
- SEP (SAPPHIRE etc.)
- External data (ODI ...)
- your model (needs to output a **particle spectra**

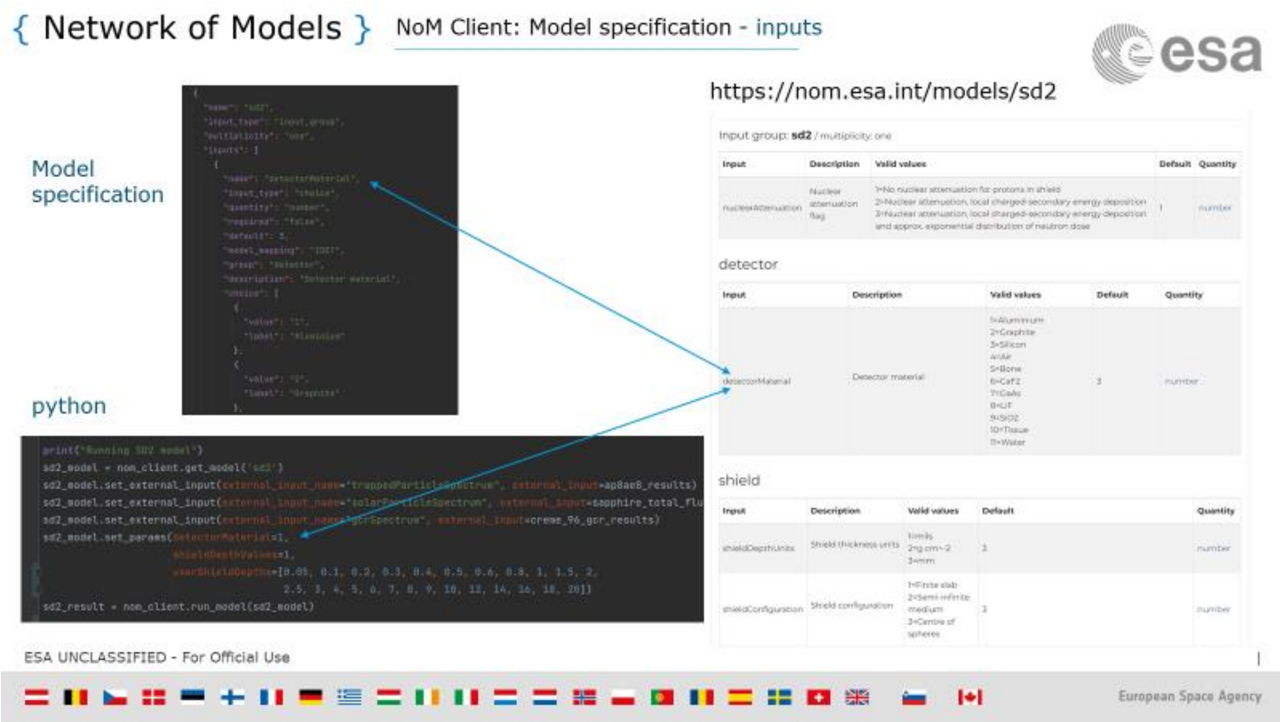
```
nom_client = NoMClient("SD2 model example", client_configuration=conf, default_server_id="nom_rest_server")

sapre_model = nom_client.get_model('sapre')
sapre_model.set_params(orbitType="GEN")
```

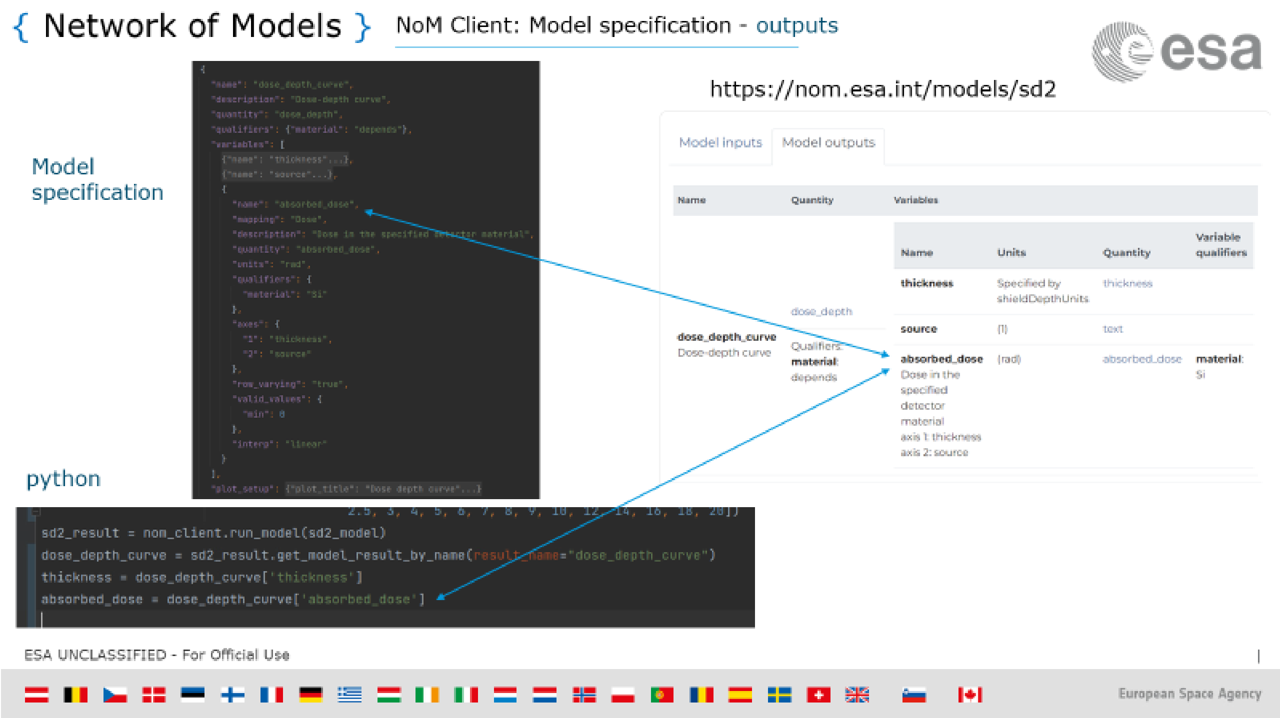




```
# Explore results
dose_depth_curve = sd2_result.get_model_result_by_name(result_name="dose_depth_curve")
thickness = dose_depth_curve['thickness'].values
absorbed_dose = dose_depth_curve['absorbed_dose'].values
```

ESA UNCLASSIFIED - For Official Use

European Space Agency


{ Network of Models }
NoM Client: Model specification - inputs
esa
https://nom.esa.int/models/sd2
Model specification
python
Input group: sd2 / multiplicity: one
detector
shield
ESA UNCLASSIFIED - For Official Use
European Space Agency


{ Network of Models }
NoM Client: Model specification - outputs
esa
https://nom.esa.int/models/sd2
Model specification
python
Model inputs
Model outputs
Name
Quantity
Variables
Units
Variable qualifiers
thickness
Specified by shieldDepthUnits
source
absorbed_dose
(rad)
material:
Si
Dose in the specified detector material
axis 1: thickness
axis 2: source
dose_depth_curve
Dose-depth curve
dose_depth
Qualifiers:
material:
depends
sd2_result = nom_client.run_model(sd2_model)
dose_depth_curve = sd2_result.get_model_result_by_name(result_name="dose_depth_curve")
thickness = dose_depth_curve['thickness']
absorbed_dose = dose_depth_curve['absorbed_dose']
ESA UNCLASSIFIED - For Official Use
European Space Agency

{ Network of Models } NoM Client: API uses

- Within SPENIVS5 the nom client API is used:
  - to provide access to the models
  - NoM model specifications are used by SPENVIS front end to dynamically generate model input GUI forms
  - Provide a framework to programmatically determine the inputs and outputs of models and how to pipeline them

Thank you for your attention!

simon.clucas@esa.int

https://nom.esa.int

Feel free to come and find me to get more information about using the ESA Network of Models

ESA UNCLASSIFIED - For Official Use

European Space Agency

Interesting case studies

- Digitalisation of complex workflows, e.g. creating environment specifications for complex, multi-segment missions
- Accessing data via ODI and seamlessly using with the full range of effects tools available
- Model/data provision layer within other systems (SPENVIS, HIERRAS, ESPREM, SRASO …)
- Leveraging Pythons machine learning libraries
- Straightforward client + web API access to legacy models