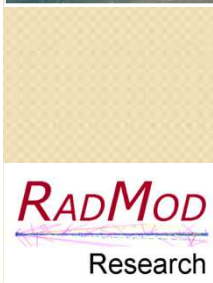


An Introduction to the REST-SIM Simulation Framework

Fan Lei
RadMod Research

(and the REST-SIM project team)



Contents

- ESA Cosmic Version 2015-2025
- The REST-SIM Project
- The REST-SIM Framework
- First Applications
- Future Developments

Cosmic Vision Missions programme

ESA's science missions for 2015-2025:

- Over 150 missions have been proposed
- 4 missions have been selected so far:
 - L class: JUICE (L1,2022)
 - M class: Solar Orbiter (M1, 2017), Euclid (M2, 2020)
 - S class: CHEOPS (S1,2017)
- Other L class candidates:
 - IXO/ATHENA: Advanced Telescope for High Energy Astrophysics
 - LISA/NGO: New Gravitational wave Observatory
- New M3 studies:
 - EChO (Exoplanet Characterisation Observatory)
 - STE-QUEST (Space-Time Explorer and Quantum Equivalence Principle Space Test)
 - MarcoPolo-R (Asteroid sample return)
 - LOFT (Large Observatory For X-ray Timing)
 - PLATO (PLANetary Transits and Oscillations of stars)

REST-SIM Project

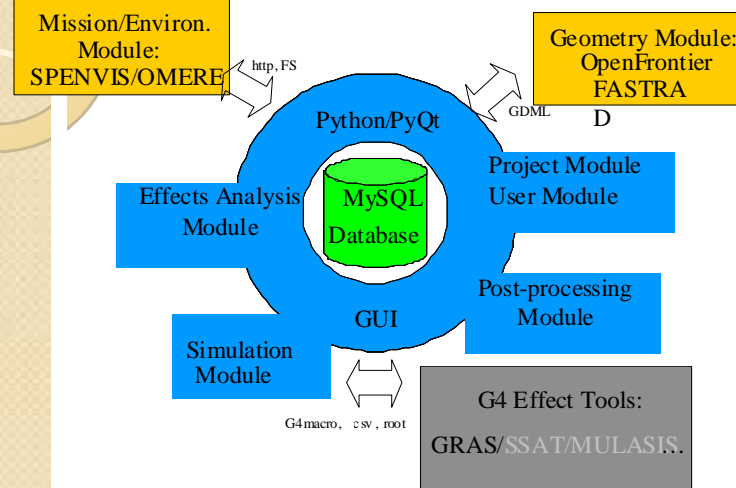
Radiation Effects on Sensors and Technologies for Cosmic Vision Missions (CVM)

- Main objectives:
 - Survey of the proposed CVM technologies and review their radiation susceptibility
 - Review existing radiation effects analysis tools and capture the SF requirements
 - Design and implement the SF
 - Demonstrate the SF capabilities
- The project team:
 - QinetiQ (now RadMod), etamax, SpacelT, DHC and UCL/CSR
 - ESA technical officers: Giovanni Santin & Petteri Nieminen
- Main development activities have been completed in 2012

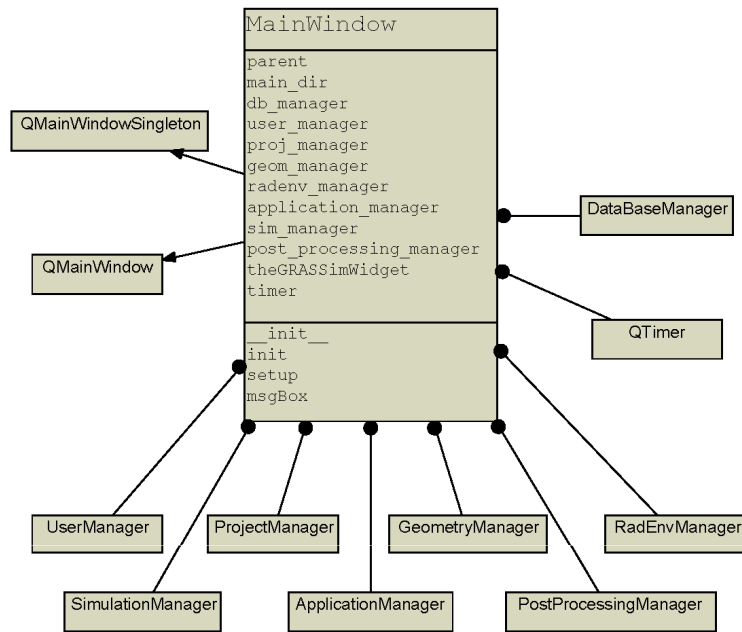
<http://spitfire.estec.esa.int/trac/REST-SIM>

REST-SIM Simulation Framework

REST-SIM Simulation Framework



Top level design



Main class diagram

Key s/w technologies:

- Python and PyQt – main programming lang. and GUI
- GRAS/Geant4 – particle transport and effects simulation tool
- OpenCascade – geometry modelling
- NumPy, SciPy & Matplotlib – post-processing
- MySQL – internal database

Main tools:

- Eclipse/Pydev, QtDesigner

REST-SIM SF Components

- **User manager:** sets up a single user (anonymous, without login) or multi user environment (with registration and login)
- **Project manager:** all configuration and run parameters, user inputs, run results, post-processing products, ... are organised in projects.
- **Environment manager:** set up connections to a SPENVIS server and import environment spectra from SPENVIS, OMERE or via direct user file import.
- **Geometry manager:** import GDML files or set up, run and import files from ESABASE2 and FASTRAD.
- **Application manager:** set up the parameters for the selected effects tool.
- **Simulation manager:** define host environments for simulation runs and schedule the run execution
- **Post-processing manager:** visualise and plot results, apply response functions and user defined functions.
- **Database:** storage of user defined parameters, imported and generated files, simulation setup and progress, ...

REST-SIM SF GUI

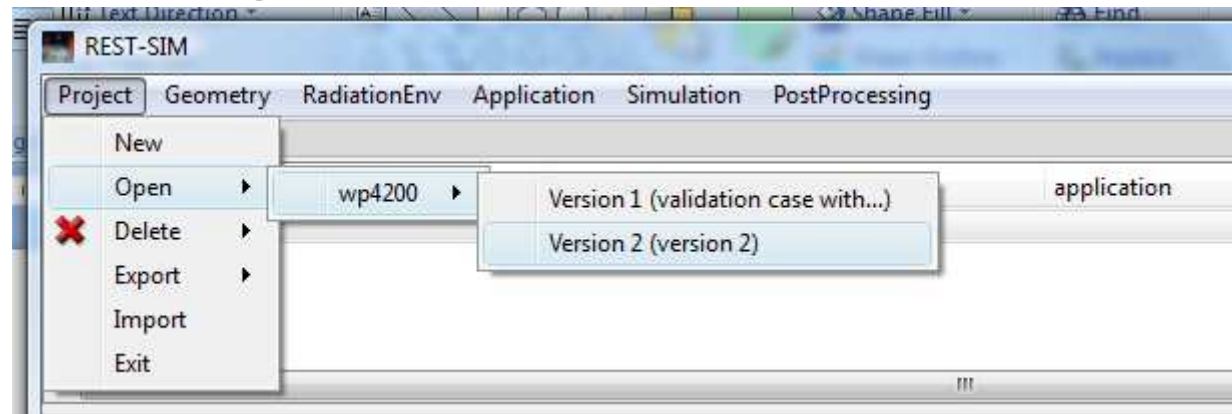
The screenshot displays the REST-SIM SF GUI interface. The main window is titled "REST-SIM" and contains several panels:

- Project Viewer:** A tree view showing the project structure. The "Post-processing" folder is expanded, showing a file named "py_test.py".
- File Editor:** A code editor showing the contents of "py_test.py". The code is as follows:

```
1 import numpy as np
2 import pylab as pl
3
4 x=2.*np.pi*np.arange(101)/100
5 pl.plot(x,np.cos(x))
6 pl.show()
```
- Post Processing:** A panel with tabs for "Results Viewer", "Python console", and "Plot Commands". The "Python console" tab is active, showing the execution of the script and the resulting array for 'x':

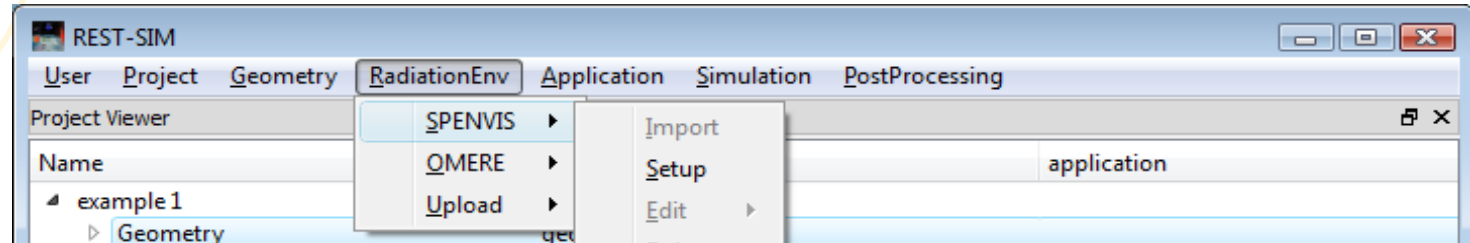
Name	Type	Size	Value
x	float64	(101,)	array([0. , 0.06283185, 0.12566371, 0.18849556, 0.25132741, ...])
- Figure 1:** A plot window showing a cosine wave. The x-axis ranges from 0 to 7, and the y-axis ranges from -1.0 to 1.0. The plot shows a single cycle of a cosine wave starting at (0, 1.0).
- GRAS Input Builder:** A panel for configuring simulation parameters. It includes a "Save" button, an "input_name" field, and a "Reset" button. The "Physics" tab is selected, and the "Partide type" is set to "e-". Other parameters like "Radius", "θmax [deg]", and "φmax [deg]" are also visible.

Project manager



- Create new and delete existing projects
- Create and delete project versions: creates a copy from a selected base version
- Share/unshare project versions with other users
- Block/unblock project versions
- Export/import projects: write/read all project files into/from a directory tree on disk
- Project viewer: GUI panel listing project version files with right-click actions: view, edit, delete, run, ...

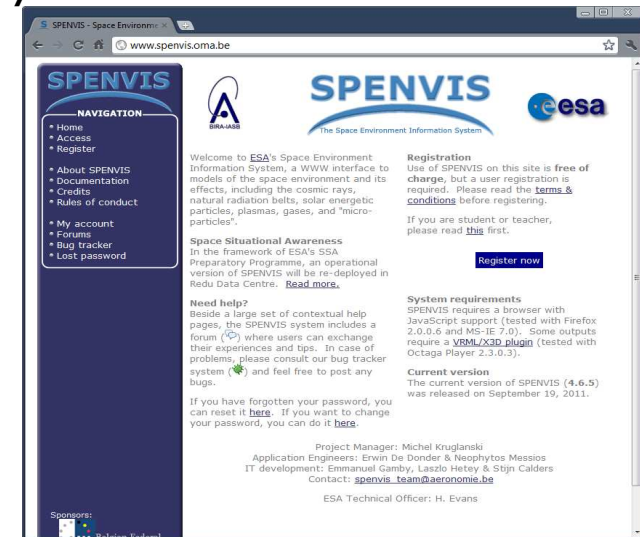
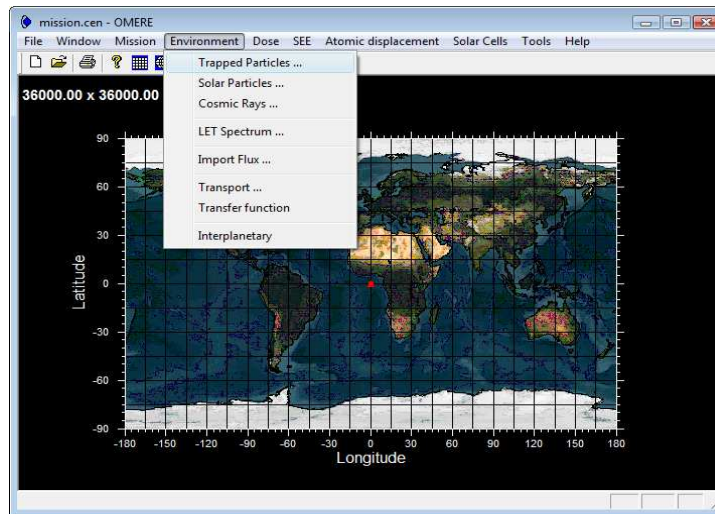
RadEnv Manager



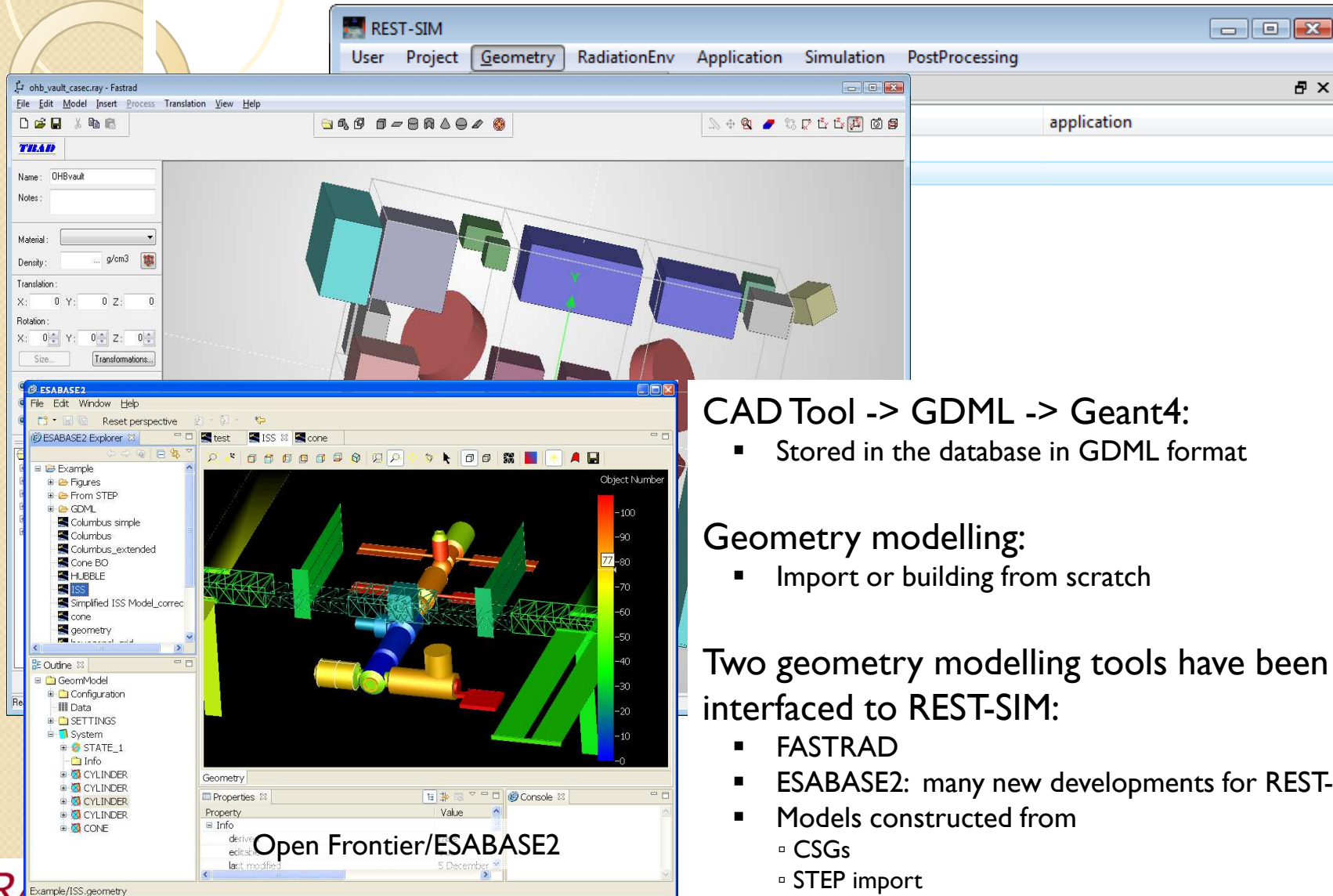
Mission environments can be modelled using SPENVIS and OMERE

- run from REST-SIM
- environ. data are imported and saved in the project database

User can also upload environ. specifications directly



Geometry Manager



CAD Tool -> GDML -> Geant4:

- Stored in the database in GDML format

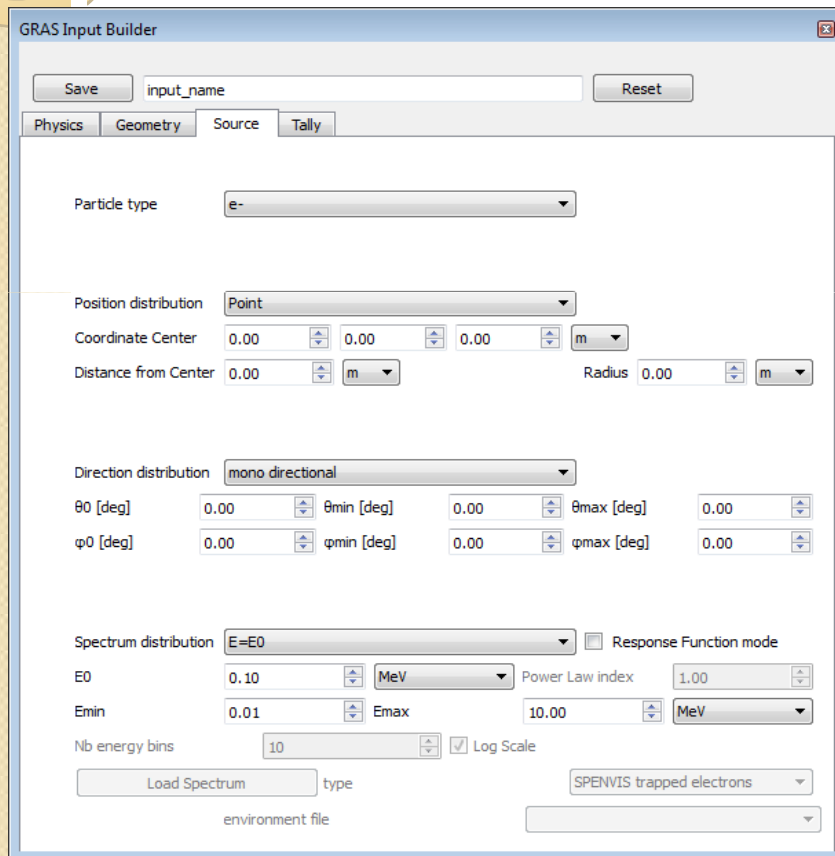
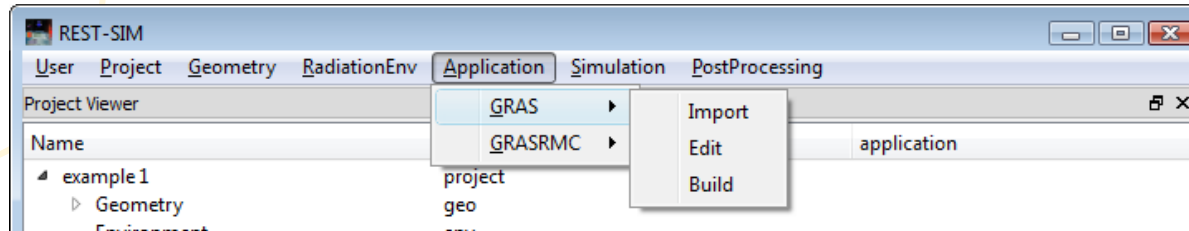
Geometry modelling:

- Import or building from scratch

Two geometry modelling tools have been interfaced to REST-SIM:

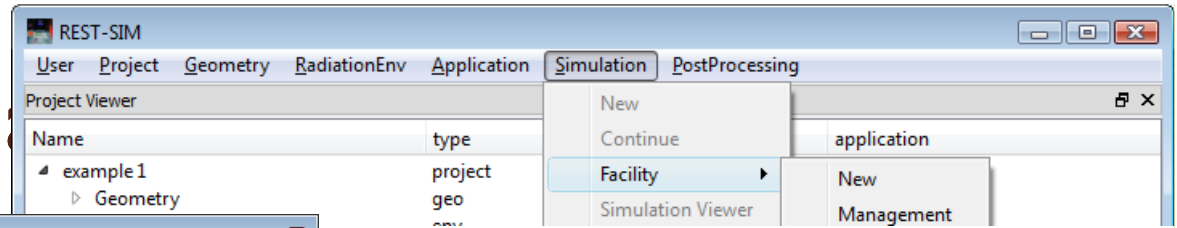
- FASTRAD
- ESABASE2: many new developments for REST-SIM
- Models constructed from
 - CSGs
 - STEP import

Application/Effects Analysis Manager



- Geant4 based analysis tools:
 - GRAS
 - GRASRMC
 - (MULASSIS, SSAT, PLANETOCOSMICS)
- Geometry and Environment from the database
- Full control of Geant4 physics
- Type of effects/analysis:
 - Fluence/Current -- Dose
 - PHS -- Dose_equivalent
 - Equivalent_dose -- LET
 - NIEL -- Path_Length
 - Charging -- Charge_collection
- Parametric and Templated analysis

Simulation Management



Simulation Facility Viewer

Name	Host name	IP	Procs	Load	Mips	Bindir	Tmpdir
Local host	['localhost']	['']	[2]	['2.0', '2.0', '2.0']	[2500]	C:\Users\flei\restsim\bin/	C:\Users\flei\res
Spitfire	['spitfire.estec.esa.int']	['']	[16]	['']	[3000]	/home/qinetiq/restsim/bin/	/home/qinetiq/

Define a simulation

New Simulation

Simulation Name:

Input XML file:

Number of Events:

Batch Execution. Nr. of runs:

Interactive runs

Node name	Host name	Nr of processors	Nr of runs
Local host	localhost	2	2
Spitfire	spitfire.estec.esa.int	16	10

- Simulation facilities:
 - Local host, or/and remote (SSH)
 - Linux, or/and Windows (local)
- Two execution modes:
 - Interactive, forced runs
 - Batch queue
- Automated parallelisation:
 - Load balance
 - Results - auto collection, merge
- Execution monitoring/management:
 - Check progress: % completed
 - Stop/Kill/Remove

Simulation Viewer

Name	host id	process id	start time	status	percentage completed	id
sim-1				running		5
run2	1	4572	2011-10-10 16:25:58	running	28.0 % completed in 102.572 s	9
run1	1	4104	2011-10-10 16:25:57	running	26.0 % completed in 97.516 s	8
sim-2				holding		6
run1	None	None	None	holding	None	10
run2	None	None	None	holding	None	11
run3	None	None	None	holding	None	12

Post-processing manager

The screenshot displays the REST-SIM software interface. The main window is titled 'REST-SIM' and has tabs for 'Project', 'Geometry', 'RadiationEnv', 'Application', 'Simulation', and 'PostProcessing'. The 'PostProcessing' tab is active, showing a 'Results Viewer' and a 'Python console'. The console displays the output of a Python script, including the version information and the creation of a NumPy array. Below the console is a 'Variables' table with the following data:

Name	Type	Size	Value
x	float64	(101,)	array([0. , 0.06283185, 0.12566371, 0.18849556, 0.25132741, ...])

Overlaid on the main window is a 'File Editor' window showing a Python script named 'py_test.py' with the following code:

```
1 import numpy as np
2 import pylab as pl
3
4 x=2.*np.pi*np.arange(101)/100
5 pl.plot(x,np.cos(x))
6 pl.show()
```

Below the code editor is a 'Figure 1' window showing a plot of a cosine wave. The x-axis ranges from 0 to 7, and the y-axis ranges from -1.0 to 1.0. The plot shows a smooth curve starting at (0, 1.0), reaching a minimum at approximately (3.14, -1.0), and returning to (6.28, 1.0).

Interactive Python scripts

- NumPy, SciPy, Matplotlib
- Python console and editor

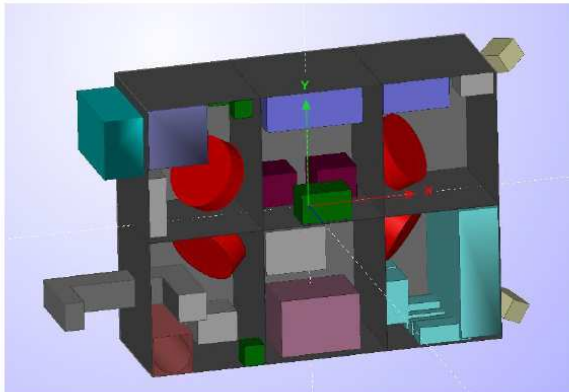
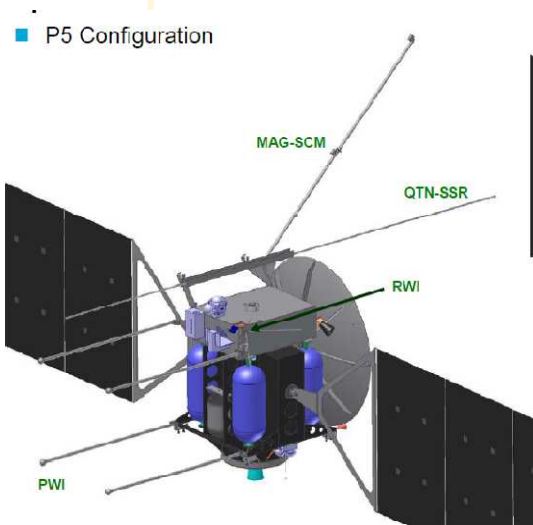
Plotting:

- 1d/2d histograms

Post-processing:

- Operation on histograms
- Derivative parameter analysis
- Analysis based on response functions
- ...

Demonstration Application: JUICE



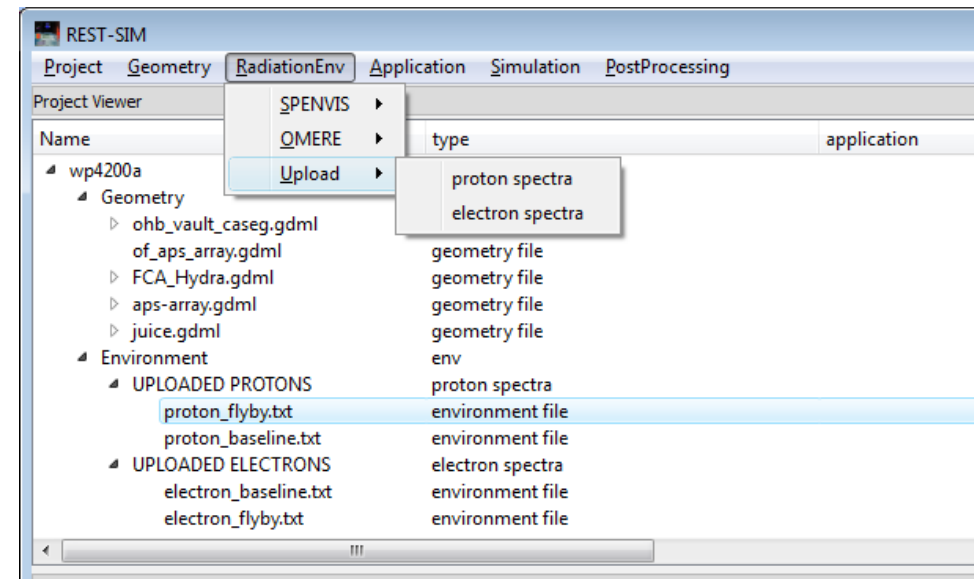
- Two demonstration applications:
 - JUICE
 - Solar Orbiter
- JUICE:
 - Environments: ESA specifications
 - Geometry Model:
 - Simplified OHB study configuration
 - Detailed geometry model of the StarTracker/APS
 - Modelled with FASTRAD
 - Analysis:
 - TID the APS/StarTracker, and others
 - Comparison with SSAT results

JUICE - Environments

- **ESA Specification:**
 - Memorandum on 'Radiation Environment Specification for Jupiter Mission Reformulation Activities' (SRE-PA/2011.050/CE issue 1.3, 10/08/2011)
- **Two mission scenarios:**
 - Baseline: Callisto + Ganymede
 - Flybys: No Callisto + 2 Europa flybys
- **Total proton and electron fluence spectra**
 - proton_baseline.txt
 - proton_flyby.txt
 - electron_baseline.txt
 - electron_flyby.txt

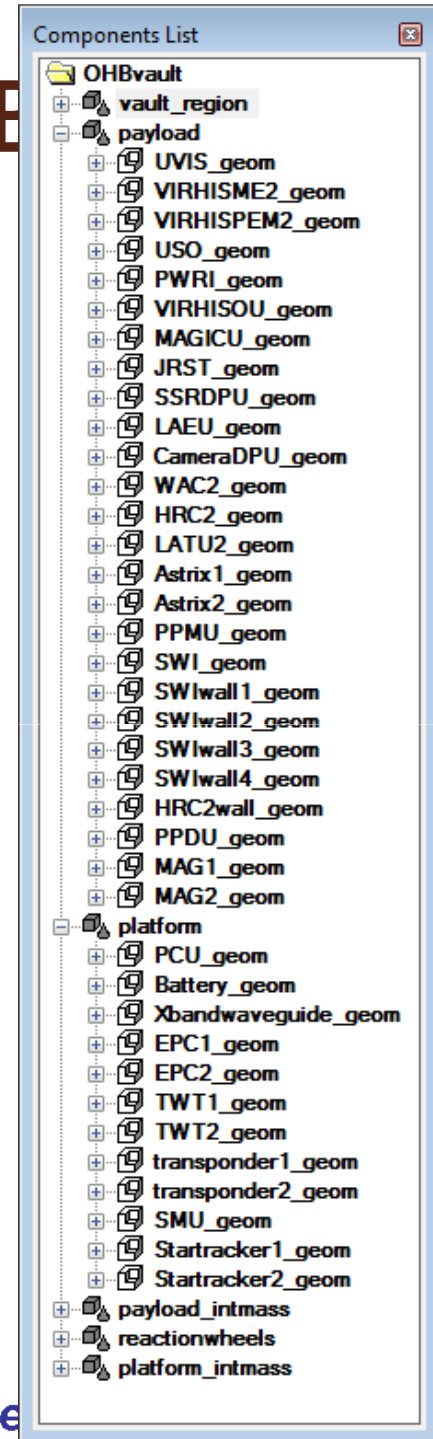
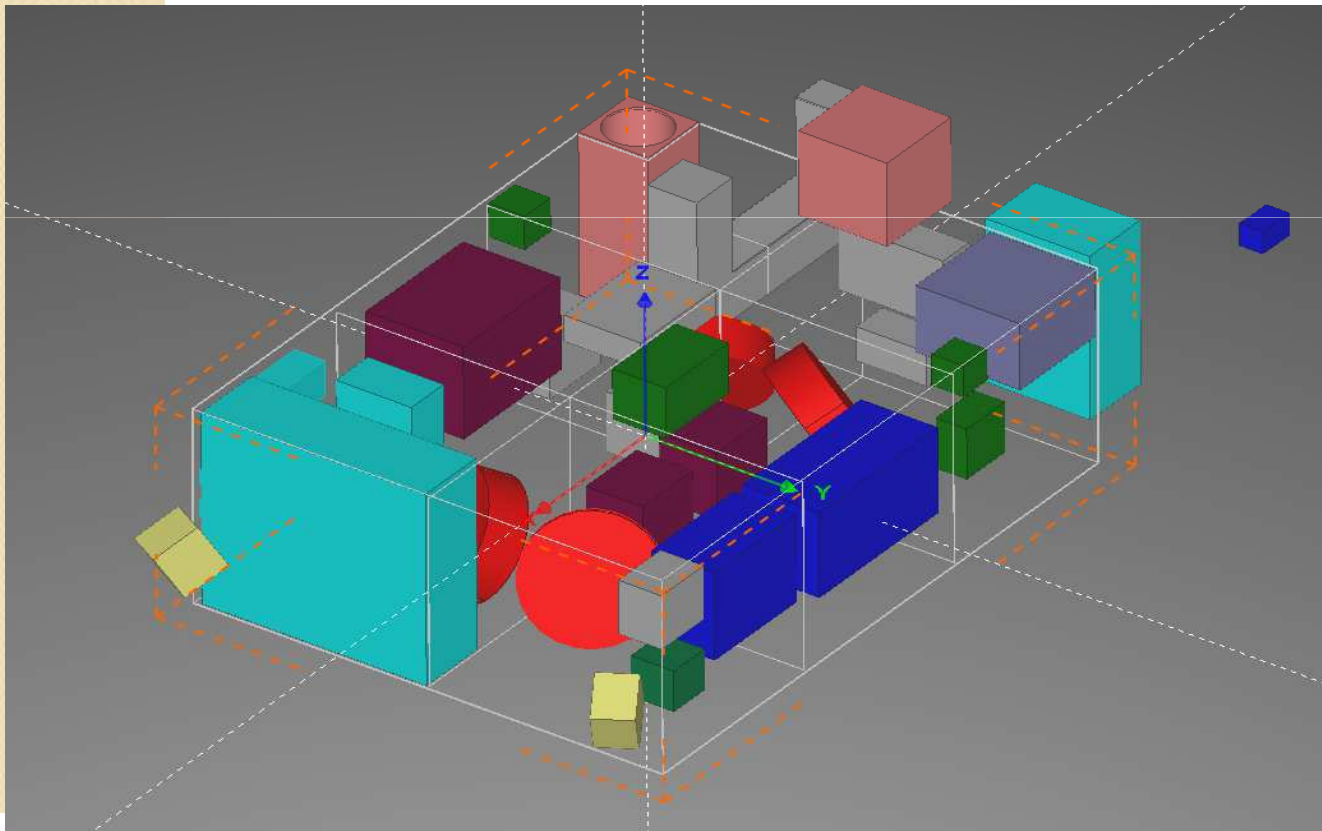
- proton_flyby.txt
- # JUICE flyby proton fluence
- # Energy spectrum

#	Energy (MeV)	Differential_flux (MeV-1.cm-2.sr-1)	Integrated_flux (cm-2.sr-1)
0.1	0.1	1.17E+14	2.48514E+13
0.2	0.2	4.79E+13	1.56082E+13
....
....
400	400	5.45E+04	7812679.029
500	500	2.34E+04	4144082.801
700	700	6.61E+03	1751316.074
1000	1000	1.57E+03	716314.2436



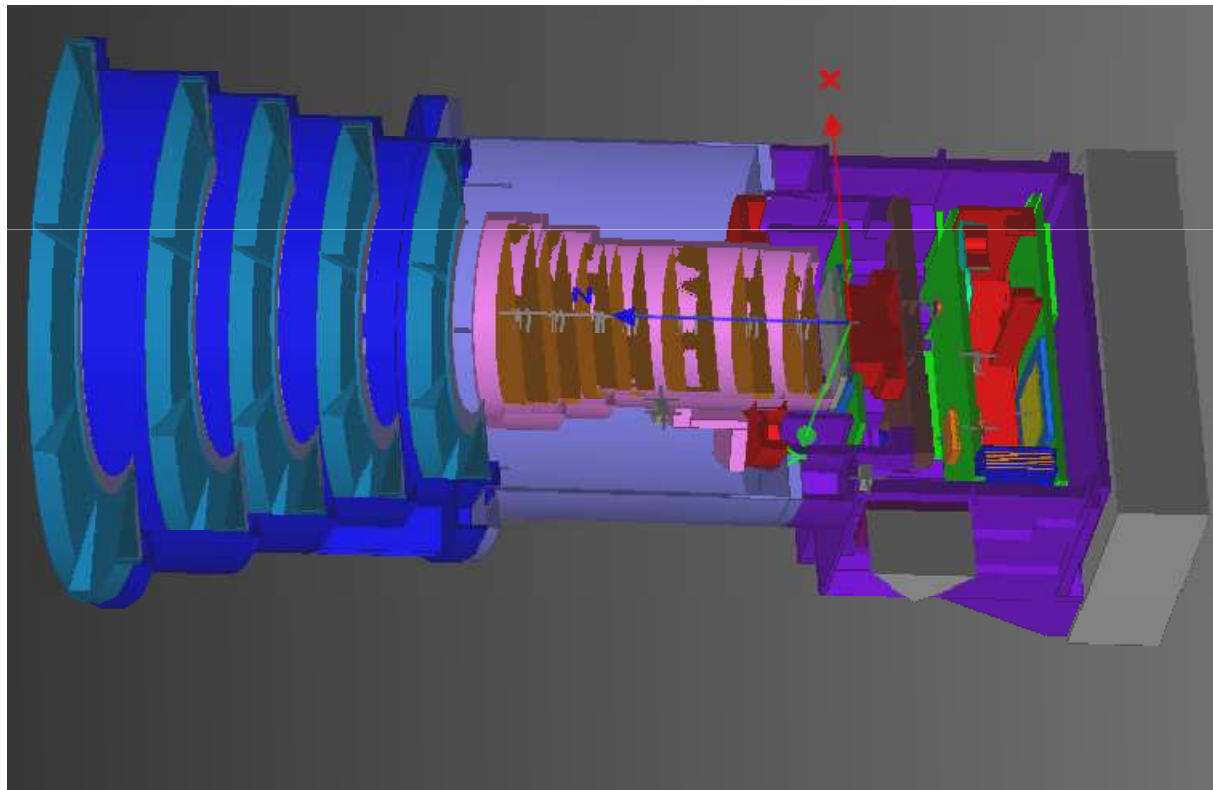
JUICE Geometry: QinetiQ/OHE

- A FASTRAD model of the service and payload modules of JUICE have been created using FASTRAD, based on the results of a separate study by OHB/QINETIQ
 - Most components are housed in a vault which is not shown

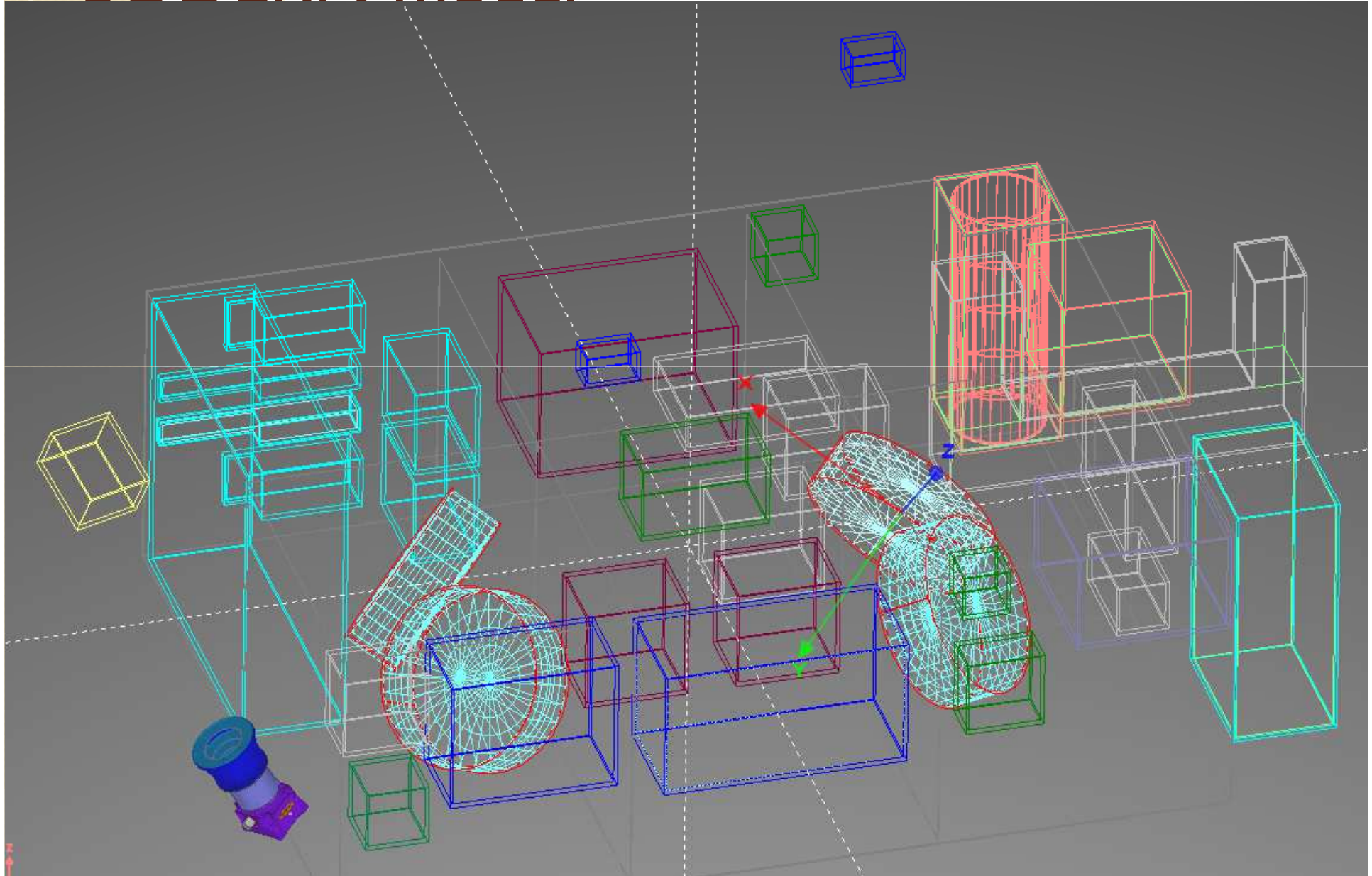


JUICE Geometry: Detailed model of the Star Tracker

A FASTRAD model of the Star Trackers has been developed by SODERN and made available to the REST-SIM project.

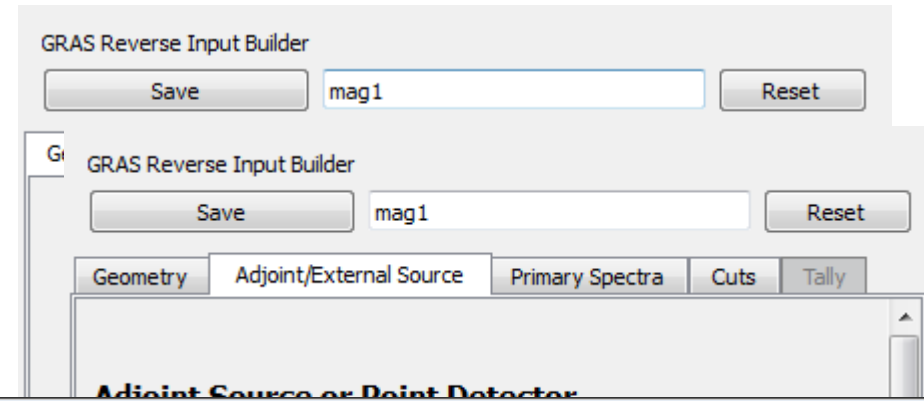


JUICE Geometry: QinetiQ/OHB + SODERN model



TID Analysis with GRASRMC

- TID analysis:
 - GRASRMC
 - QinetiQ/OHB geometry model
 - Both baseline and flyby environments
 - For 10 different units in the payload and service modules



Platform/Payload

PCU
 TWT #1
 SMU
 SG ACS STS AA STR
 SG ACS STS AA STR
 UVIS2
 VIRHIS OU (outside vault)
 SWI (outside vault)
 PP DU (outside vault)
 MAG #1 (outside vault)

Name	type	application	date	id
wp4200a	project			13
▶ Geometry	geo			
▶ Environment	env			
▶ Simulations				
Post-processing	analysis			
▶ Input	input			
twl.nml	namelist file	GRASRMC	2012-04-28 19:36:16	2016
smu.nml	namelist file	GRASRMC	2012-04-28 19:40:57	2018
swi.nml	namelist file	GRASRMC	2012-04-28 18:56:40	1989
stk2.nml	namelist file	GRASRMC	2012-04-28 19:50:32	2022
mag1.nml	namelist file	GRASRMC	2012-04-28 19:03:41	1991
ppdu.nml	namelist file	GRASRMC	2012-04-28 19:52:52	2024
<u>uvis2.nml</u>	namelist file	GRASRMC	2012-04-28 19:51:32	2023
virhis.nml	namelist file	GRASRMC	2012-04-28 18:34:35	1965
hydra_d1.nml	namelist file	GRASRMC	2012-05-04 11:17:28	2158
aps-array.nml	namelist file	GRASRMC	2012-04-29 00:39:25	2128
a-a-e1.nml	namelist file	GRAS	2012-04-29 00:48:47	2140
of_array.nml	namelist file	GRAS	2012-04-29 22:37:54	2150
pcu.nml	namelist file	GRASRMC	2012-04-28 19:35:08	2014
stk1.nml	namelist file	GRASRMC	2012-04-28 19:50:22	2021



Simulation executions

Define a simulation

New Simulation

Simulation Name:

Input XML file:

Project Viewer

Name	type	application	date
wp4200a	project		
Geometry	geo		
Environment	env		
Simulations			
ppdu	simulation	GRASRMC	2012-04-28 21:03:07
ppdu1335643387_1212287075_Spectrum3.csv	csv file	GRASRMC	2012-04-28 21:25:07
ohb_vault_caseg.gdml	gdml file	GRASRMC	2012-04-28 21:25:07
ohb_vault_caseg_materials.xml	xml file	GRASRMC	2012-04-28 21:25:07
ppdu1335643387_1212287075.csv	csv file	GRASRMC	2012-04-28 21:25:07
ppdu1335643387_1212287075Convergence_TID.txt	txt file	GRASRMC	2012-04-28 21:25:07
ppdu1335643387_1212287075_Spectrum1.csv	csv file	GRASRMC	2012-04-28 21:25:07
ppdu1335643387_1212287075_Spectrum2.csv	csv file	GRASRMC	2012-04-28 21:25:07
ppdu.nml	namelist file	GRASRMC	2012-04-28 19:52:52
ppdu1335643387_1212287075_Spectrum4.csv	csv file	GRASRMC	2012-04-28 21:25:07
ppdu_1335643387_1212287075.g4mac	g4mac file	GRASRMC	2012-04-28 21:25:07
ppdu_1335643387_1212287075.g4mac.log	log file	GRASRMC	2012-04-28 21:25:07
swi	simulation	GRASRMC	2012-04-28 18:57:16
stk1	simulation	GRASRMC	2012-04-28 19:55:15
virhis	simulation	GRASRMC	2012-04-28 18:36:25
uvis2	simulation	GRASRMC	2012-04-28 21:02:33

GRASRMC Simulation Results

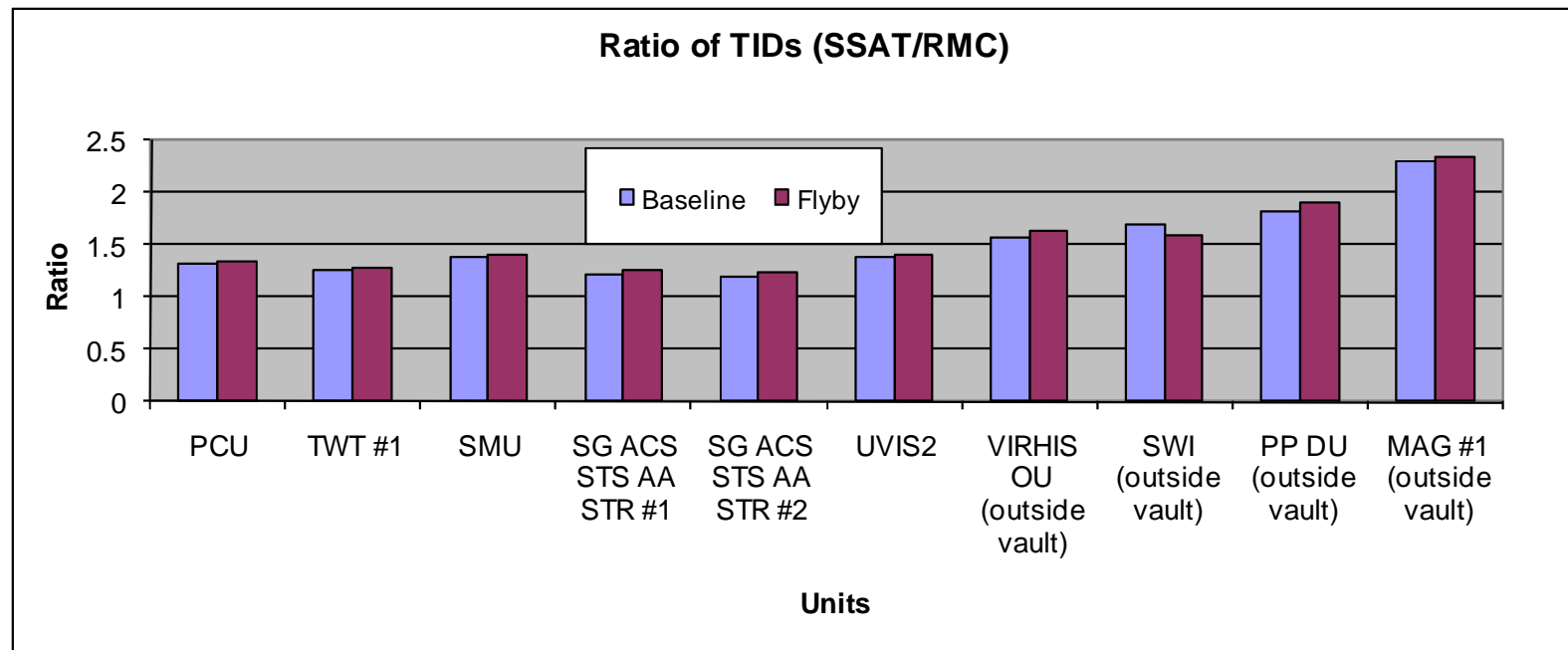
		baseline			flyby		
		electron	proton	total	electron	proton	total
Platform	PCU	3.03±0.01	67.1±3.1	70.1±3.1	3.04±0.01	81.1±4.1	84.1±4.1
	TWT #1	3.07±0.01	71.1±3.1	74.1±3.1	3.12±0.01	86.1±3.1	89.1±3.1
	SMU	3.01±0.01	63.1±2.1	66.1±2.1	3.05±0.01	78.1±1.1	81.1±1.1
	Star Tracker 1	3.82±0.02	88.1±2.1	92.1±2.1	3.81±0.02	107.1±2.	111.1±2.1
	Star Tracker 2	3.78±0.02	90.1±2.1	93.1±2.1	3.83±0.02	109.1±2.	112.1±2.1
Payload	UVIS2	3.17±0.04	62.1±1.1	65.1±1.1	3.2±0.01	76.1±2.1	80.1±2.1
	VIRHIS OU	3.38±0.02	124.1±4.1	127.1±4.1	3.42±0.02	142.1±4.	145.1±4.1
	SWI	3.62±0.02	129.1±13.	133.1±13	3.66±0.02	153.1±15	156.1±15.
	PP DU	2.15±0.01	43.1±2.1	45.1±2.1	2.18±0.01	53.1±3.1	55.1±3.1
	MAG #1	4.23±0.03	67.1±1.1	71.1±1.1	4.3±0.03	84.1±1.1	88.1±1.1

TID in krad(Si)

Comparison with SSAT Results

- SSAT analysis were carried out for the same locations with the same geometry model. The calculated TID in krad(Si)

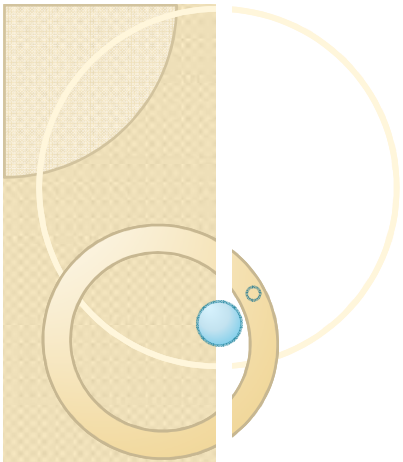
		Baseline	Flybys
Platform	PCU	90.69	112.83
	TWT #1	92.27	113.53
	SMU	90.83	113.03
	Star Tracker 1	111.52	138.06
	Star Tracker 2	111.2	138.17
Payload	UVIS2	89.53	111.64
	VIRHIS OU	197.39	237.55
	SWI	225.53	248.87
	PP DU	82.52	103.52
	MAG #1	163.01	204.89



Future Developments

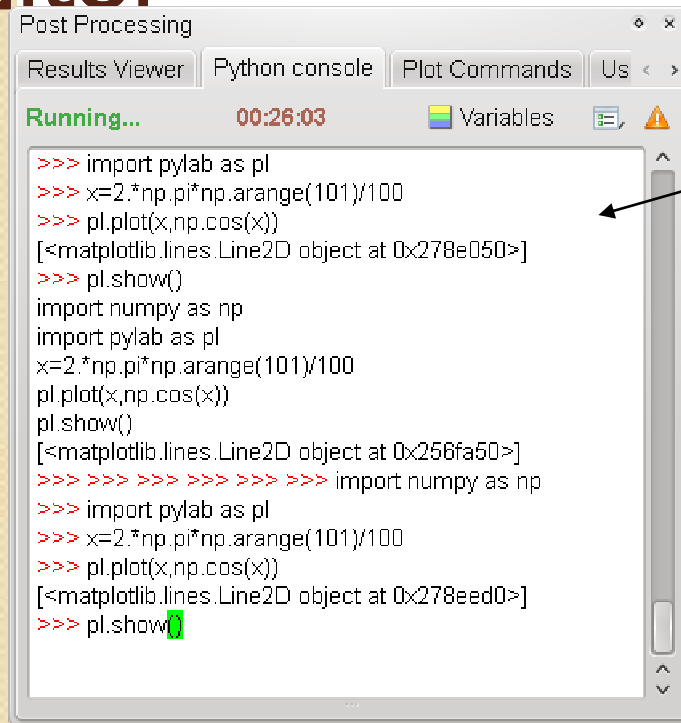
A new team has been assembled in response to new ESA ITT (CIRSOS):

- **New Integrated Modelling Environment (IME)**
 - collaborative and iterative modelling approach, operational s/w for CVMs
- **Geometry manager**
 - Model configuration tool: GDML – modular schema
 - New visualisation tool
- **Application tools**
 - Internal Charging (IC) analysis
 - Based on the ELSHIELD work
 - Build-in libraries of materials and components at risk
 - SSAT
 - 2-stage analysis approach
 - FMC and RMC in both stage
 - General parametric analysis
 - Template based solution
- **Simulation manager**
 - Utility to use commercial cloud computing facilities, e.g. EC2
- Lots of enhancement to post-processing and many more...



Backup slides

Interactive Python console/ Python editor



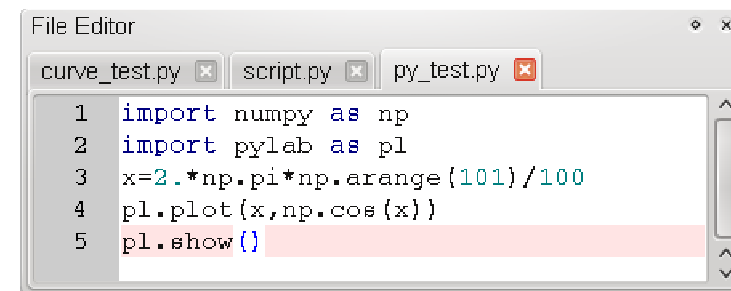
```
>>> import pylab as pl
>>> x=2.*np.pi*np.arange(101)/100
>>> pl.plot(x,np.cos(x))
[<matplotlib.lines.Line2D object at 0x278e050>]
>>> pl.show()
import numpy as np
import pylab as pl
x=2.*np.pi*np.arange(101)/100
pl.plot(x,np.cos(x))
pl.show()
[<matplotlib.lines.Line2D object at 0x258fa50>]
>>> >>> >>> >>> >>> >>> import numpy as np
>>> import pylab as pl
>>> x=2.*np.pi*np.arange(101)/100
>>> pl.plot(x,np.cos(x))
[<matplotlib.lines.Line2D object at 0x278eed0>]
>>> pl.show()
```

Interactive Python console

- Using the Spyder python library
- Run of user script from popup menu of the PYTHON editor
- Visualisation of python command send by the GUI
- Pre imported modules : numpy, pylab, DataManager, function loading

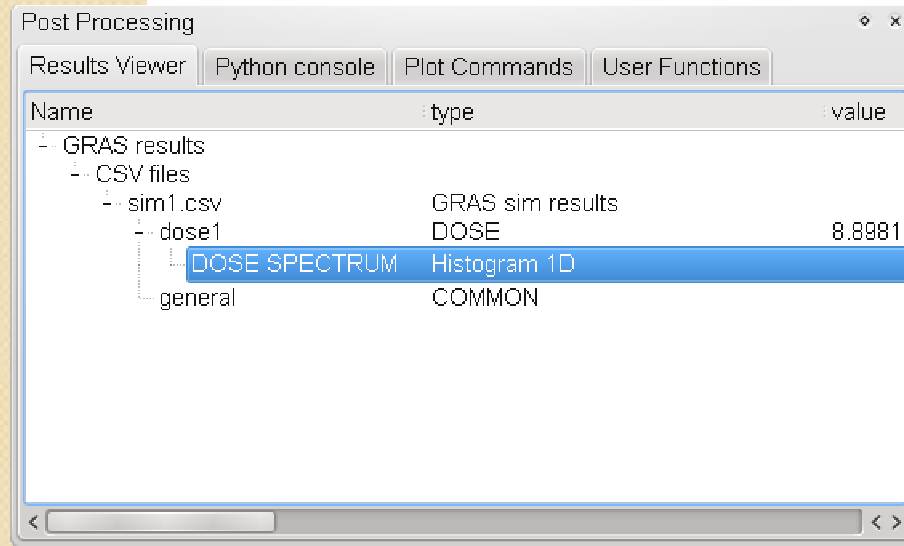
Python script editor

- Syntax recognition based on QScintilla
- menu for run, save, edit, add commands



```
1 import numpy as np
2 import pylab as pl
3 x=2.*np.pi*np.arange(101)/100
4 pl.plot(x,np.cos(x))
5 pl.show()
```

Reading/Viewing/Access of simulation results



Reading/Viewing of simulation results

- PYTHON interface to Spenvis CSV C++ code to read CSV files
- DirectView of GRAS CSV file in Editor
- Results viewer widget as a file structure viewer

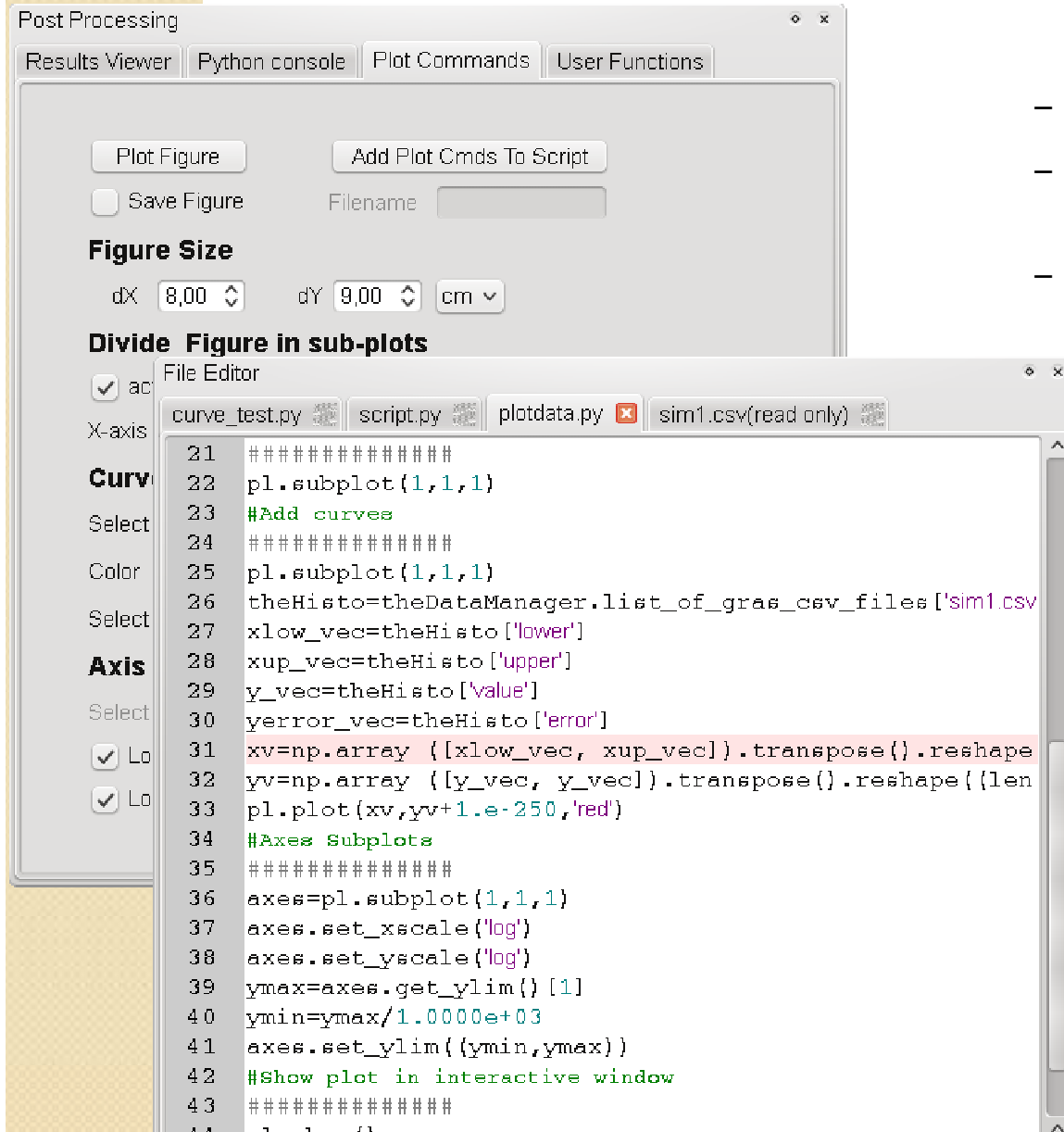
Access of data in user script

- Access of DataManager in user script
- Python Cmds to access Histograms directly added in the script from a popup menu of the results viewer

The screenshot shows a "File Editor" window with several tabs: "curve_test.py", "script.py", "data.py", and "sim1.csv(read only)". The "data.py" tab is active, showing the following Python code:

```
2 theDataManager=AnalysisDataManager ()
3 #Load GRAS csv file sim1.csv
4 theDataManager.load_gras_csv_file(115,data_name='sim1.csv')
5 theHisto=theDataManager.list_of_gras_csv_files ['sim1.csv'] ['histo1D']
6 xlow_vec=theHisto ['lower']
7 xup_vec=theHisto ['upper']
8 y_vec=theHisto ['value']
9 yerror_vec=theHisto ['error']
```

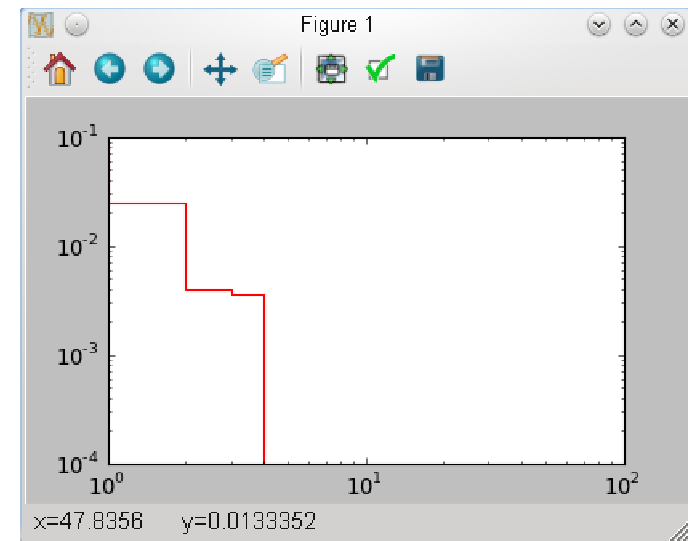
Plotting of simulations results



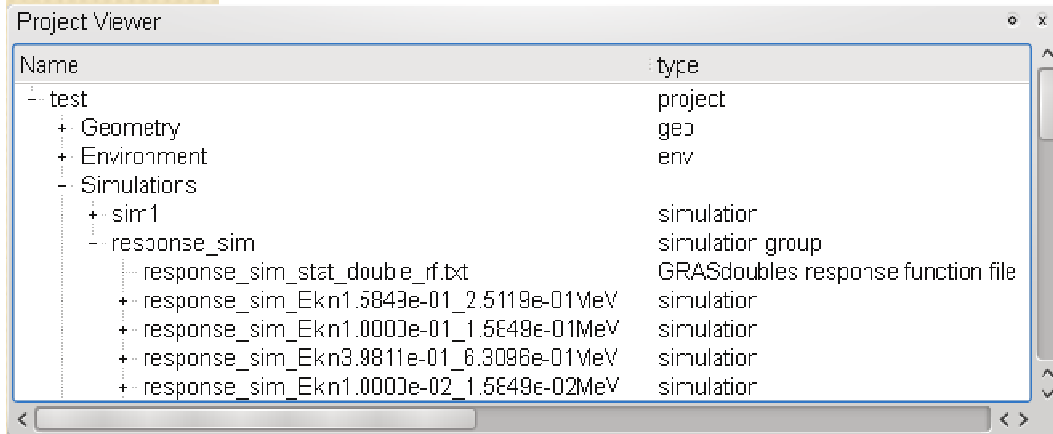
The screenshot shows the 'Post Processing' window with the 'Plot Commands' tab selected. Below the tabs are buttons for 'Plot Figure', 'Add Plot Cmds To Script', 'Save Figure', and a 'Filename' input field. The 'Figure Size' section has 'dX' set to 8,00, 'dY' to 9,00, and 'cm' as the unit. The 'Divide Figure in sub-plots' section has a checked checkbox. The 'File Editor' shows a script named 'plotdata.py' with the following code:

```
21 #####
22 pl.subplot(1,1,1)
23 #Add curves
24 #####
25 pl.subplot(1,1,1)
26 theHisto=theDataManager.list_of_gras_csv_files['sim1.csv']
27 xlow_vec=theHisto['lower']
28 xup_vec=theHisto['upper']
29 y_vec=theHisto['value']
30 yerror_vec=theHisto['error']
31 xv=np.array([xlow_vec, xup_vec]).transpose().reshape(1,-1)
32 yv=np.array([y_vec, y_vec]).transpose().reshape(1,-1)
33 pl.plot(xv,yv+1.e-250,'red')
34 #Axes Subplots
35 #####
36 axes=pl.subplot(1,1,1)
37 axes.set_xscale('log')
38 axes.set_yscale('log')
39 ymax=axes.get_ylim()[1]
40 ymin=ymin/1.0000e+03
41 axes.set_ylim((ymin,ymax))
42 #Show plot in interactive window
43 #####
44 pl.show()
```

- Direct plotting of single ID histogram
- Multi curve plottings using a plotting widget
- Interactive adding of plotting cmds from plotting widget to user script



Response function analysis

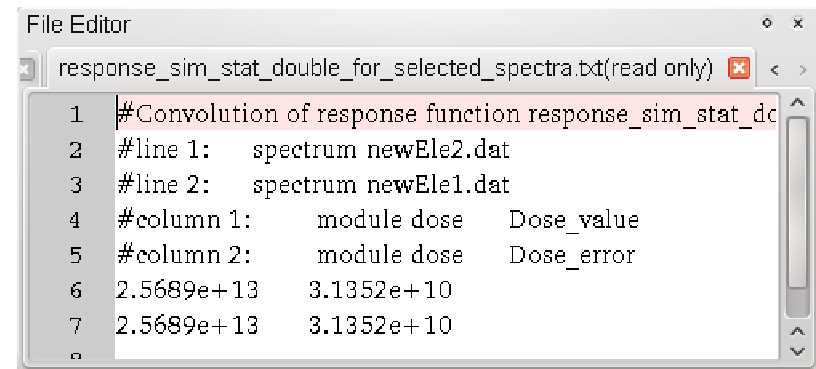


• Signals vs primary spectrum

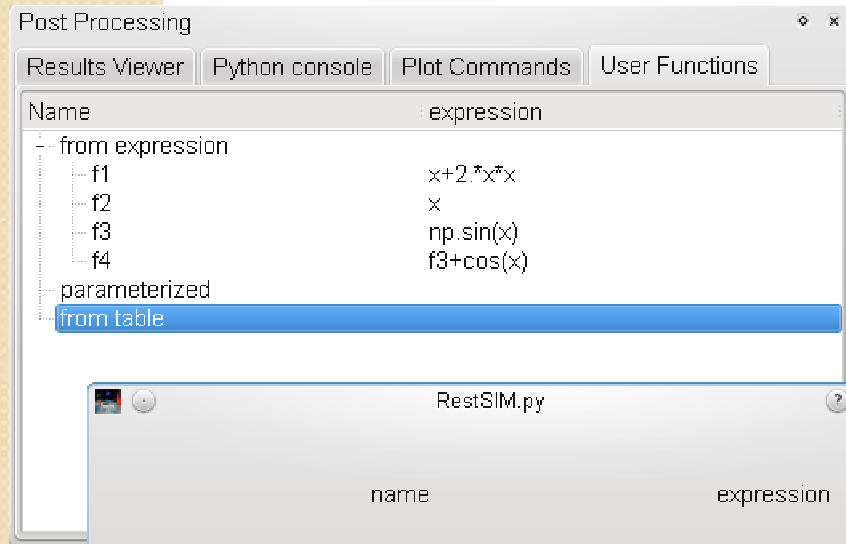
- Convolution of response functions with user selected spectra
- Use of SciPy library
- ASCII table of scalar signals in function of user selected spectra

Response functions vs primary energy

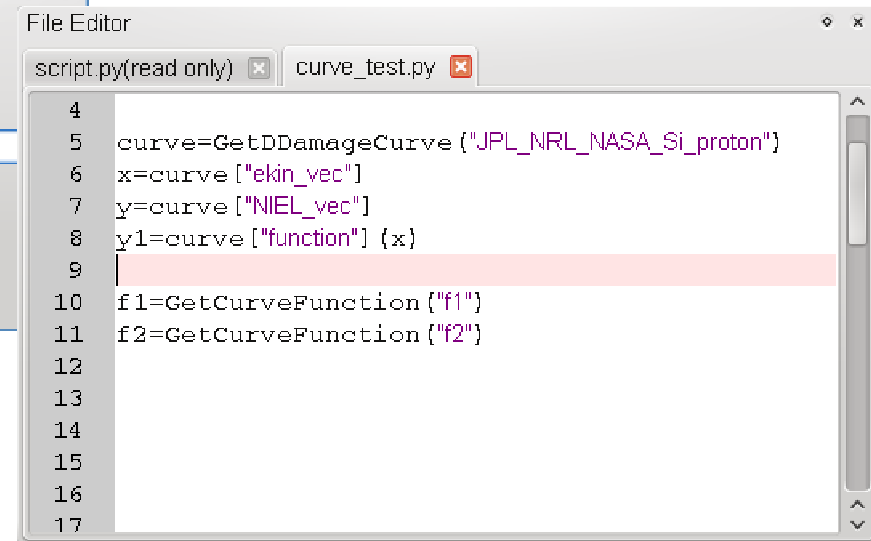
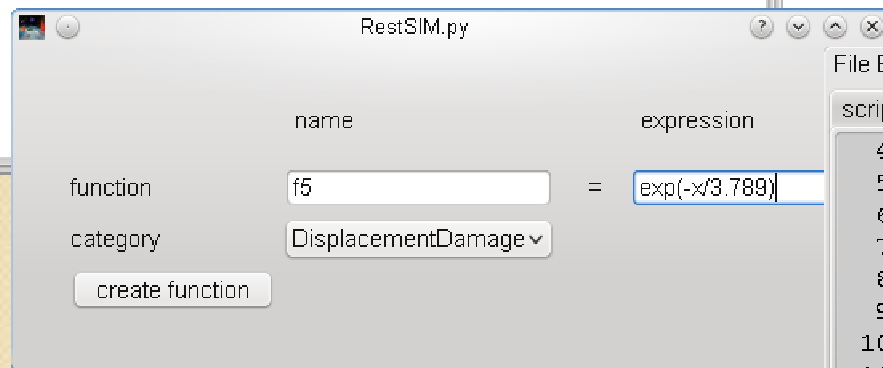
- Multiple simulations vs primary energy
- Log or linear energy bins
- General concept of simulation group for all type of parametrized simulations
- End of simulations: production of an ASCII table containing all scalars (TID, NIEL,...) in function of primary energy bins



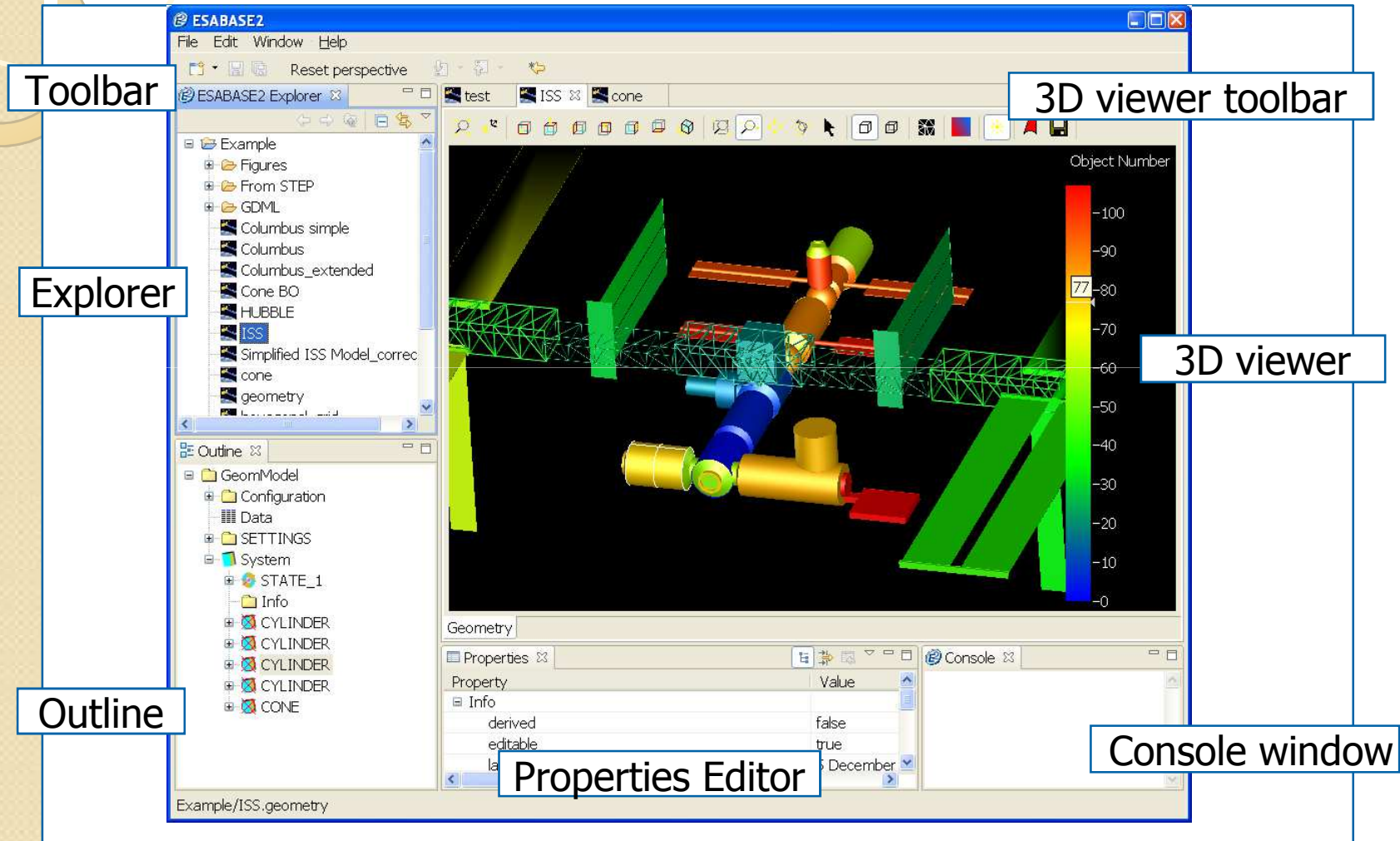
Definition/loading of user functions for radiation effect analysis



- Definition of user functions from the GUI or in PYTHON script
- User functions stored in the database
- Access of user functions in PYTHON scripts

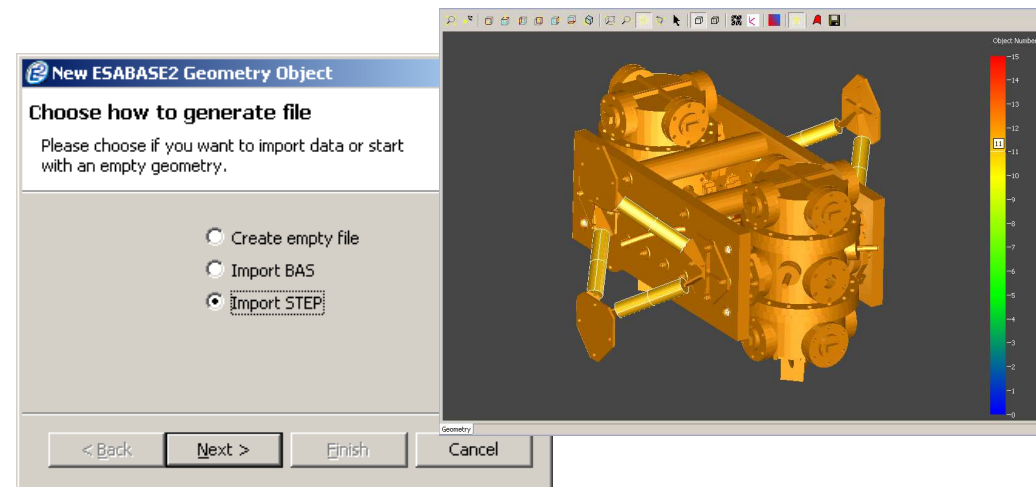
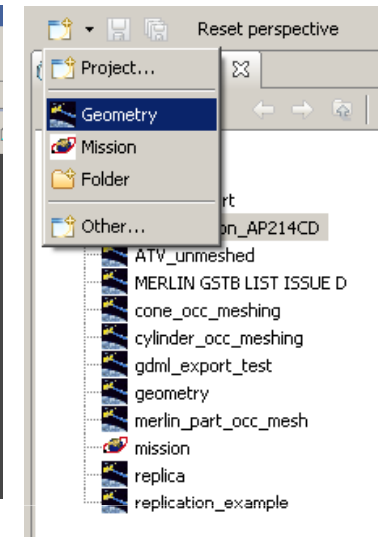
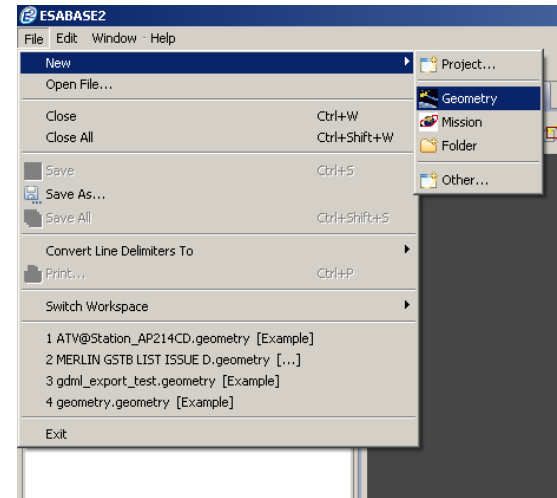


ESABASE2 GUI Overview



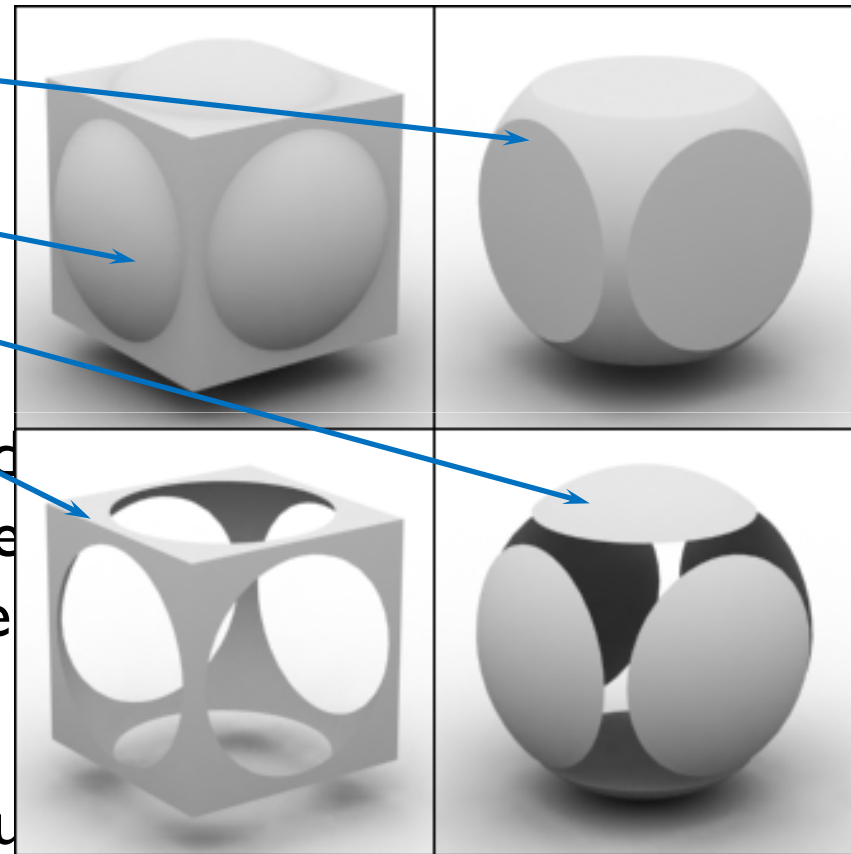
STEP AP203/214 Import

- Import of any STEP AP203/214 files, also of high complexity.
- Tested with a large number of files originating from various CAD tools.
- Accessible via the respective open file menu or toolbar entry.
- Opens a file selection dialogue
- Can take a while, depending on the complexity of the geometry...



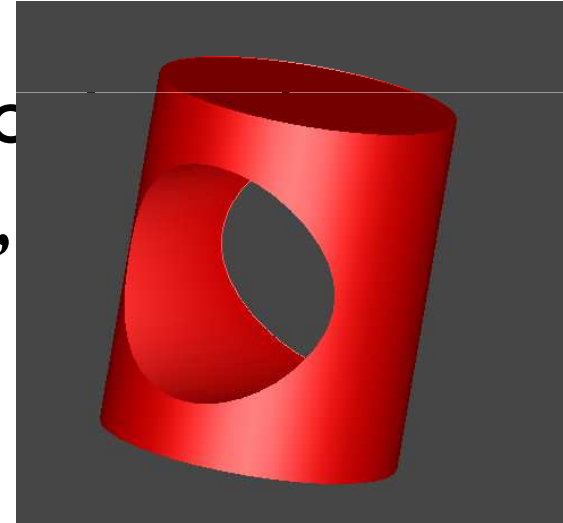
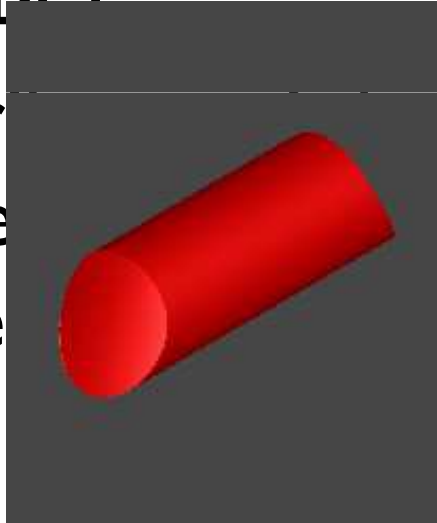
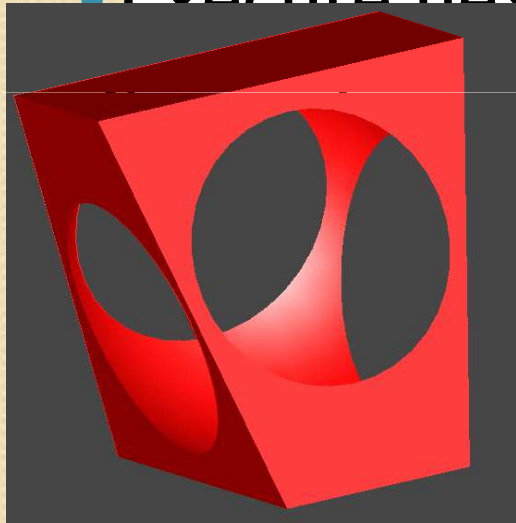
Boolean Operations (I)

- Three types of Boolean Operations (BO) implemented:
 - intersection
 - union
 - subtraction
- After the operation is performed, the child shape is stored under the BOP-node
- It can be edited via the shape
- Any changes made are applied to the Boolean shape.
- If a BO is removed from the BOP node, the operation is undone



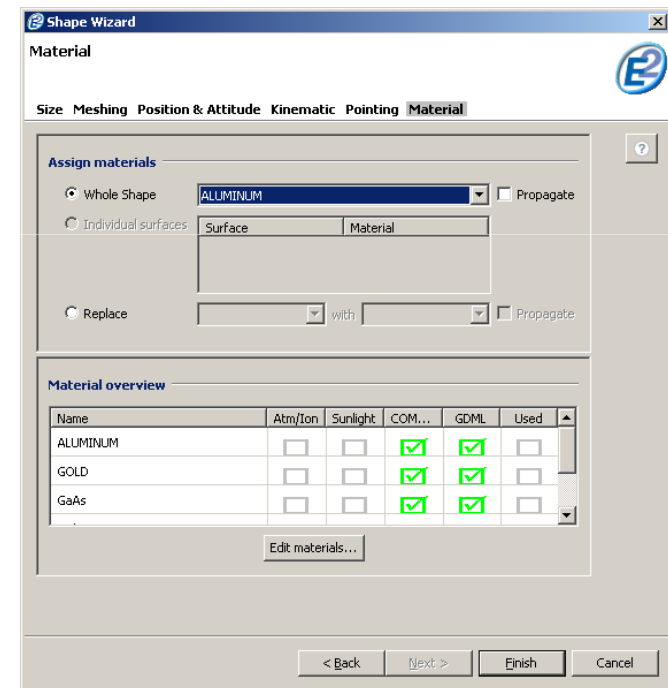
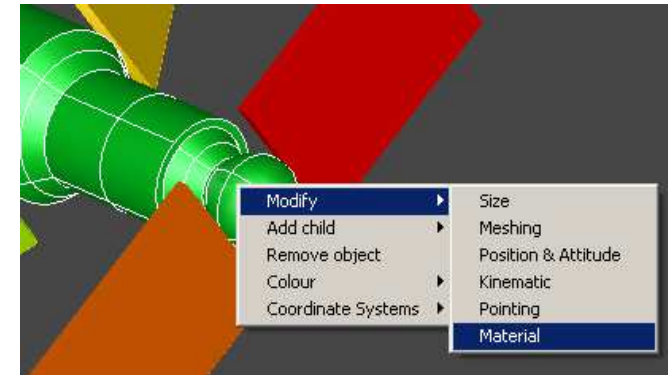
Boolean Operations (2)

- Series of Boolean Operations can be performed following the defined workflow:
- Add child to Boolean shape parent
- Execute next BO



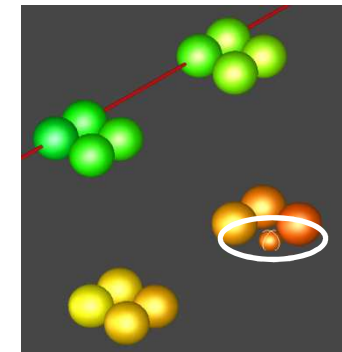
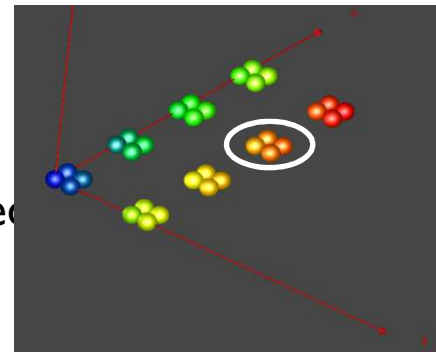
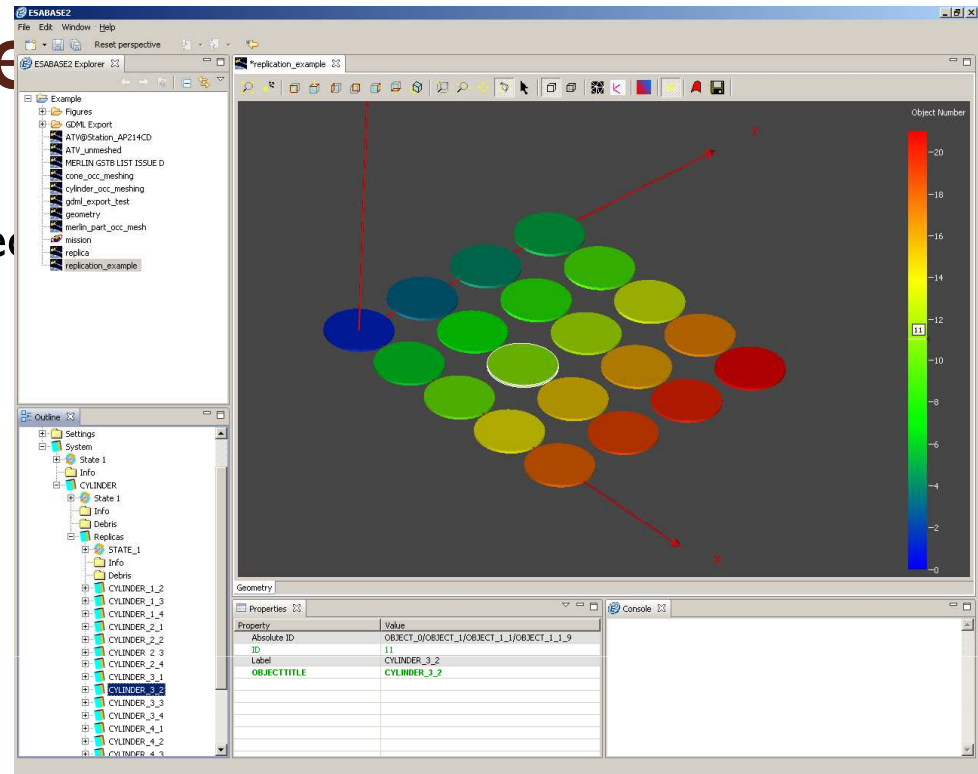
Material Editor

- Objective: Assignment of materials to geometrical objects.
- Underlying comprehensive material database.
- Part of the shape wizard.
- Accessible via
 - the Outline context menu,
 - the 3D view context menu ('Modify --> Material').
- Allows
 - editing of material properties,
 - creation of user defined materials.
- Materials can be assigned only, if they are copied from the material database to the geometry.
- The elements a material is consisting of are automatically copied to the geometry, when the material is copied to the geometry.



Replicated Structure

- Objective: enable the replication of geometrical objects in x- and y-direction with user defined different off-sets.
 - Example:
 - replicated cylinders
 - 5 in x-direction
 - 4 in y-direction
 - different off-sets in x- and y-direction
 - index of the replica is given in its name
 - here: *cylinder_3_2* :
 - 3rd in x-direction
 - 2nd in y-direction
 - Replication of the parent object of a replicated structure replicates the entire structure.
 - The properties each object of the replicated structure can be edited individually.



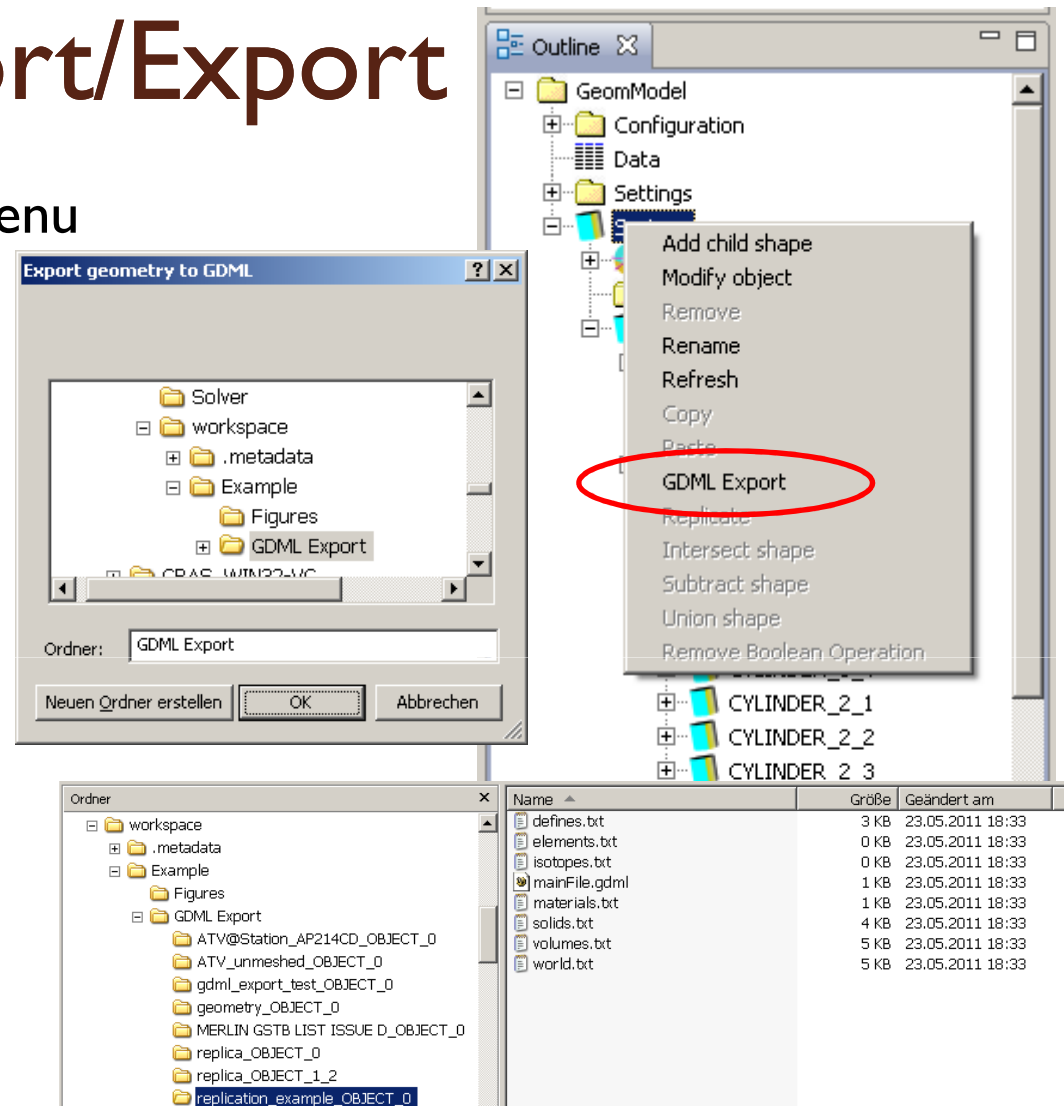
GDML Import/Export

- via the Outline context menu

- entire system
- parts of the geometry tree
- single objects

- opens a dialogue for the selection of the target folder

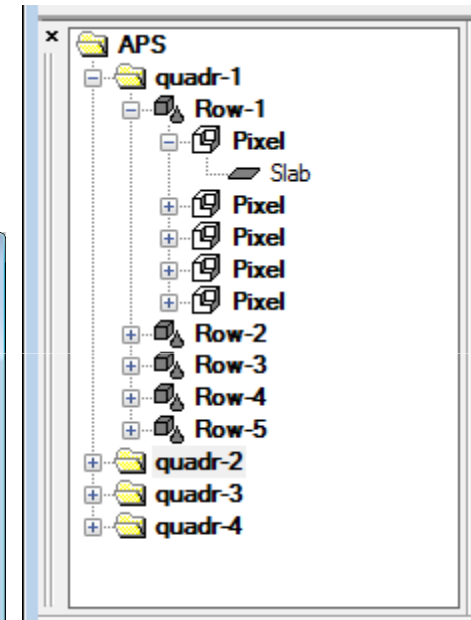
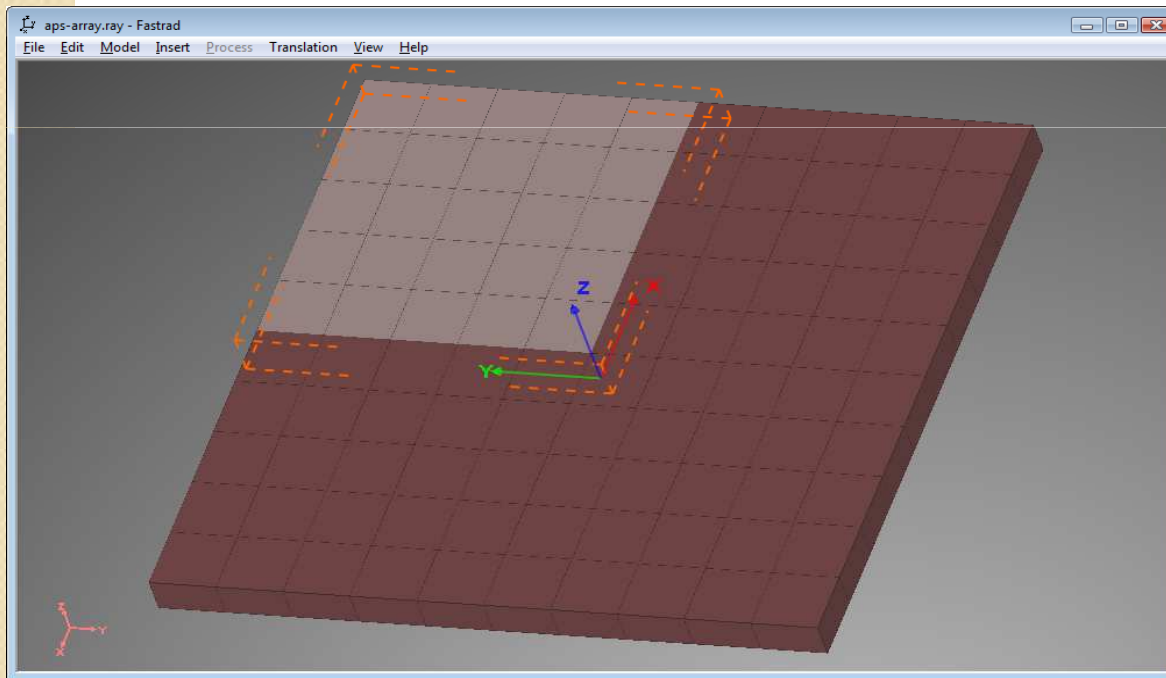
- creates a sub-folder in the target folder with the name of the geometry file, which contains the 'main.gdml' file and the include files



JUICE Geometry: Pixellated APS Model

- FASTRAD

- Repeated APS pixels can be modelled using the combined functions of copy/paste and transformation
- But volume names have to be changed manually
 - This is required if each pixel needs to be identified uniquely
 - Impractical for large arrays such as the APS (1024x1024)



JUICE Geometry: Pixellated APS Model – Open Frontier

