openSF

A Generic Framework for End-To-End Mission Performance Simulations

J.J. Ramos
Senior Software Engineer
Deimos Space S.L.

Table of contents

1. Introduction

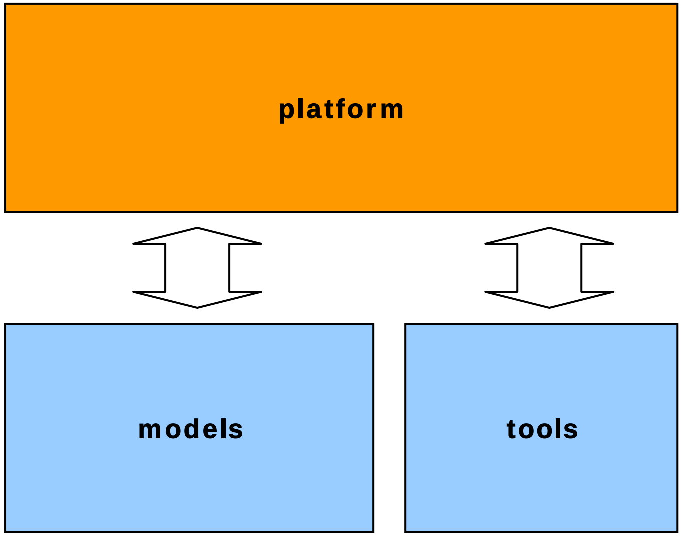
- In the frame of concept and feasibility studies for the Earth Observation (EO) activities, mission performance in terms of final data products needs to be predicted by means of so-called end-to-end (E2E) simulators.
- A specific mission E2E simulator is able to reproduce all significant processes and steps that impact the mission performance and gets simulated final data products.

1. Introduction

- openSF started as a result of the development of the E2E simulator for EarthCARE, ECSIM (ESA's contract number 20003/065/NL). This achievement has been obtained thanks to the joint work with ESA, covering their requirements and suggestions.
- openSF is a generic simulation framework product being developed by Deimos Space, S.L. aimed to cope with these major goals. It provides end-to-end simulation capabilities that allow assessment of the science and engineering goals with respect to the mission requirements.
- openSF represents a robust and extremely user-friendly tool easily adaptable to cope with any mission requirements.

2. Concept

- Independent framework providing added-value functionalities to scientific simulations.
  - Substitute complex and rigid shell scripts or Simulink projects.
- Scientific models and product exploitation tools can be plugged in the system with ease.

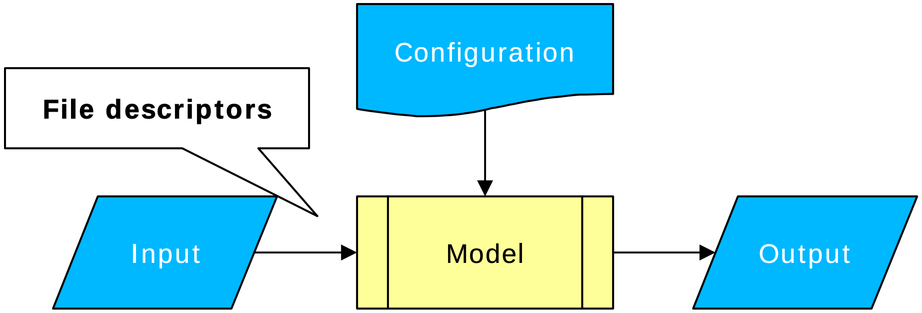


3. Simulation environment

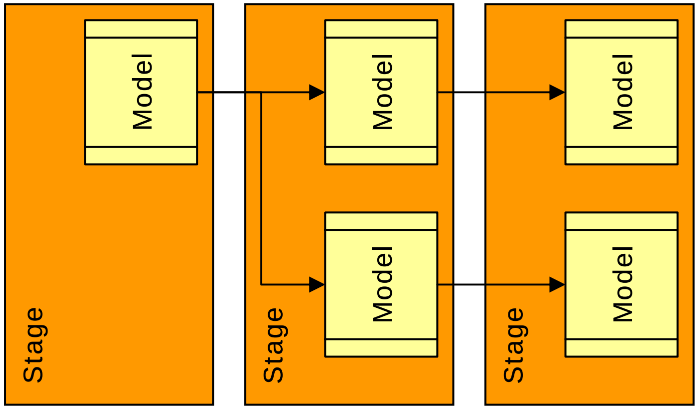
- openSF manages the three aspects of a simulation environment:
  - Static structure
  - Dynamic flow
  - Data creation and exploitation

3. 1. Static structure

- Simulations are constituted by a hierarchical structure of building blocks including the independent algorithms (models) and their relationships (interfaces or “file descriptors”).
  - Models
  - File descriptors
  - Stages/families
  - Simulations
- This modular architecture allows reusing and substituting any part of the simulation.



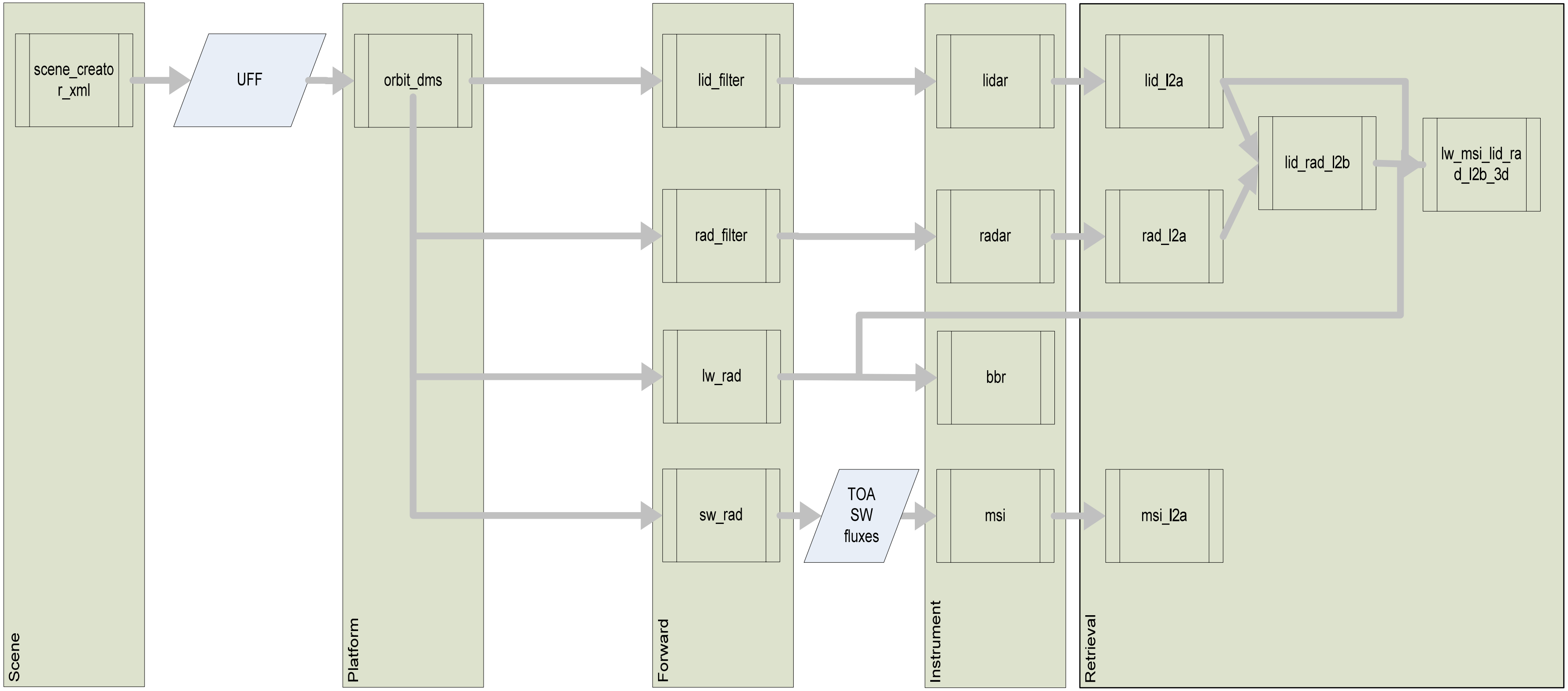
Model definition including input/output and configuration files



Simulation definition

3. 1. Static structure

- Simple and complex simulation chains can be defined.
- Can define different partitions to focus in branches (e.g. to simulate instrument losses), or stages.

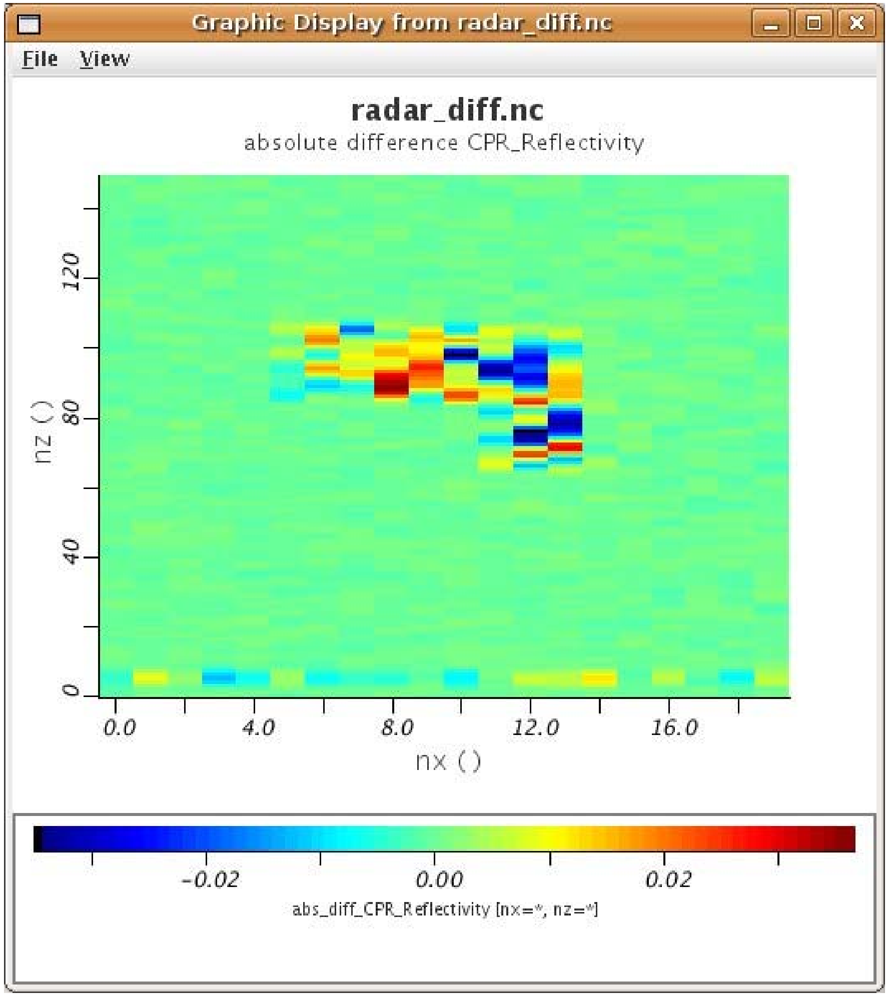


3. 2. Dynamic flow

- Controls the simulation process and data flow.
- Operators can provide with input and auxiliary files the simulation definitions.
- Performs a validation check prior to the execution.
- Runs the program in an independent process, isolating the execution environments.
- Intercepts logging messages from models
  - Information, debug, warning, error and progress messages
- Ensures error handling
  - Intercepts unexpected crashes
- Maintains an execution archive to recover past simulations.

3. 3. Data creation and exploitation

- A plug-in mechanism lets to include specific external tools or GUIs for data handling:
  - Managing data and define the input contents.
  - Examining the output with tools for plotting, browsing, importing and exporting products, etc.
- Provides a view of the local file-system for quickly access to product files and tools.


Graphic Display from radar_diff.nc
File
View
radar_diff.nc
absolute difference CPR_Reflectivity
0
40
80
120
nz ()
0.0
4.0
8.0
12.0
16.0
nx ()
-0.02
0.00
0.02
abs_diff_CPR_Reflectivity [nx=*, nz=*]

Example of external product tool (ncBrowse)

4. Documentation

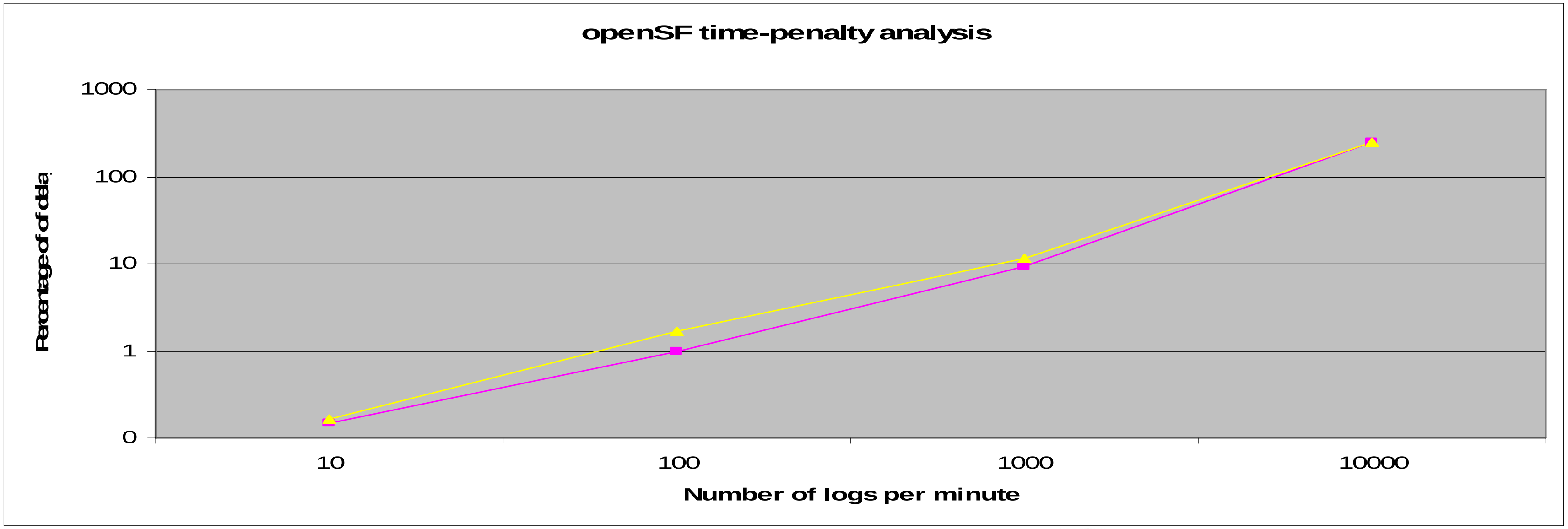
- Development will include independent documents:
  - Detailed Design Document for developers
  - Complete testing
  - Interface Control Document
  - Software User Manual
- This documentation can be added to other developments projects.

5. 1. Open

- openSF uses a flexible licensing scheme that allows integrating any kind of third-party developments.
- Core library is distributed under the GNU Lesser General Public License or LGPL that is a free software license published by the Free Software Foundation.
- LGPL places copy-left restrictions on the program itself but does not apply these restrictions to other software that merely links with the program.
- Models and tools components can use ANY license schema.
- Other components with non-compatible licenses can be requested as “pre-requisites”.

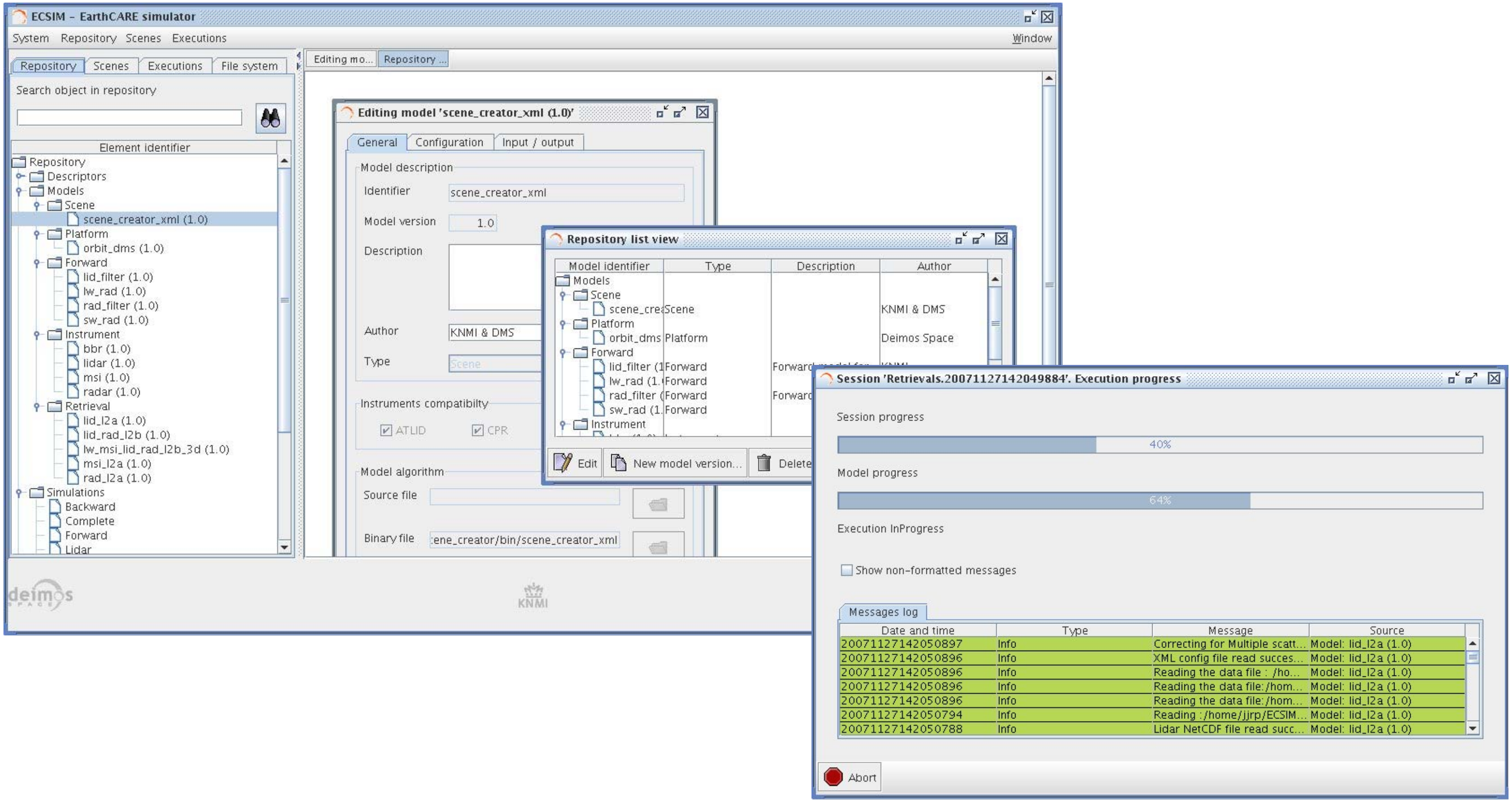
5.2. Portable

- Thanks to technology used, openSF is multi-platform with minimal hardware and software requirements:
  - Java(TM) 2 Runtime Environment, Standard Edition 1.5 or superior
  - MySQL client and server 5 or superior and MySQL connector/J 5.0.4
  - XML technologies
- Works in PC and Mac machines. 32 and 64 bits.
- Under several operating systems:
  - Windows™ OS family.
  - Linux distributions: RedHat, openSUSE, Debian, Ubuntu, …
  - Mac™ OS family.
- openSF system does not penalize in excess memory (around 30Mb) and speed performances of a scientific models.



5.3. User-friendly

- openSF provides a graphical Human-Computer Interface and a command line interface for interacting with the system.



Search object in repository

Repository
Descriptors
Models
Scene
scene_creator_xml (1.0)
Platform
orbit_dms (1.0)
Forward
lid_filter (1.0)
lw_rad (1.0)
rad_filter (1.0)
sw_rad (1.0)
Instrument
bbr (1.0)
lidar (1.0)
msi (1.0)
radar (1.0)
Retrieval
lid_l2a (1.0)
lid_rad_l2b (1.0)
lw_msi_lid_rad_l2b_3d (1.0)
msi_l2a (1.0)
rad_l2a (1.0)
Simulations
Backward
Complete
Forward
Lidar

Editing model 'scene_creator_xml (1.0)'

General | Configuration | Input / output

Model description

Identifier: scene_creator_xml

Model version: 1.0

Description:

Author: KNMI & DMS

Type: Scene

Instruments compatibilty

☑ ATLID ☑ CPR

Model algorithm

Source file:

Binary file: :ene_creator/bin/scene_creator_xml

Repository list view

Model identifier	Type	Description	Author
Models			
Scene			
scene_crea	Scene		KNMI & DMS
Platform			
orbit_dms	Platform		Deimos Space
Forward			
lid_filter (1	Forward	Forward model for	KNMI
lw_rad (1.	Forward		
rad_filter (	Forward	Forward	
sw_rad (1.	Forward		
Instrument			

Session 'Retrievals.20071127142049884'. Execution progress

Session progress: 40%

Model progress: 64%

Execution InProgress

□ Show non-formatted messages

Messages log

Date and time	Type	Message	Source
20071127142050897	Info	Correcting for Multiple scatt...	Model: lid_l2a (1.0)
20071127142050896	Info	XML config file read succes...	Model: lid_l2a (1.0)
20071127142050896	Info	Reading the data file : /ho...	Model: lid_l2a (1.0)
20071127142050896	Info	Reading the data file:/hom...	Model: lid_l2a (1.0)
20071127142050896	Info	Reading the data file:/hom...	Model: lid_l2a (1.0)
20071127142050794	Info	Reading :/home/jjrp/ECSIM...	Model: lid_l2a (1.0)
20071127142050788	Info	Lidar NetCDF file read succ...	Model: lid_l2a (1.0)

5.4. Configurable

- openSF lets users to alter its behavior and structure by implementing well-defined XML interfaces.
  - Parameters are specified by configuration files
- A database supports the system definition.
- System configuration includes management of environment variables.

```
<?xml version="1.0" encoding="UTF-8"
    standalone="no"?>
<exampleFile>
  <group1>
   <group2>
     <parameter name=“string" description="text"
     type="STRING" value=“value" />
     <parameter name=“int array"
     description="text" type="INTEGER" ndims="2"
     dims="3 3" value="1 2 3 4 5 6 7 8 9" min="2"
     max="5" />
       <group3>
         <parameter name=“float"
     description="text"   type="FLOAT" value="1.0"
     units="s" min="0" max="59" />
       </group3>
       <parameter name=“file" description="text"
     type=“FILE" value=“afile.cnf" />
   </group2>
  </group1>
</exampleFile>
```

5.5. Expandable

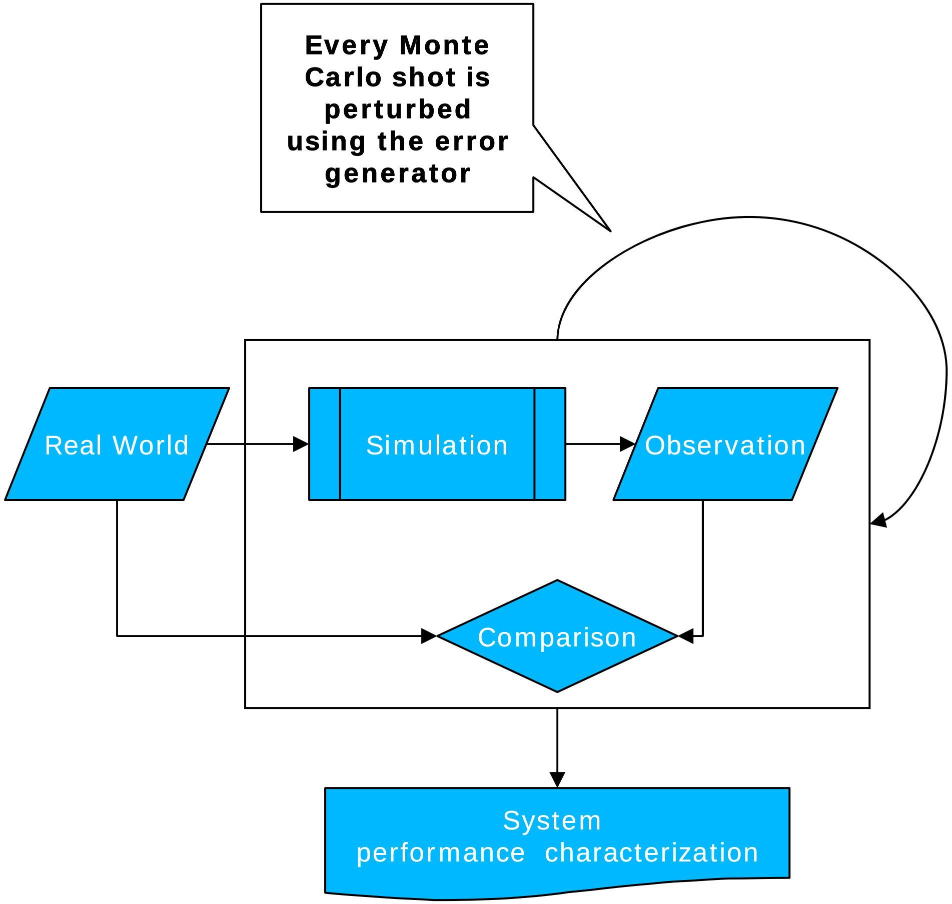
- **Users can integrate every kind of executables (no programming language dependency) as models and product tools.**
- The Interface Control Document (ICD) describes the integration requirements and some development guidelines.
- **These requirements imply a minimal intrusion into code of scientific models.**
- openSF provides some Integration libraries to ease the model integration process.

5.6. Powerful

- Users can execute simulations in batch or iterating through different parameter values.
  - Following aritmethic sequences and imported from third-party mathematical tools.
- Users can abort the execution at will and restart/resume it from last succesful stage.

5.7. Statistical simulations

- openSF will provide a way to apply statistical methods (like Monte Carlo simulations) for system performance analysis.
- Some libraries for errors definition and generation are being produced. Error sources as:
  - Polynomial functions (bias, affine, linear, parabolic, ...)
  - Random distributions (beta, gamma, exponential, normal, ...)
  - Harmonic functions
  - Step functions
  - Sampled with linear, polynomial or sp-line interpolation.
  - Plus simple arithmetical operations to combine them
- Performance analysis tools to graphically characterize the system.


Every Monte Carlo shot is perturbed using the error generator
Real World
Simulation
Observation
Comparison
System performance characterization

6. Applications

- openSF has been successfully used in some ESA-funded projects, notably:
  - ECSIM, an E2E system performance simulator for the EarthCARE mission
  - Processing chain prototypes for Earth-Observation Optical missions (GERSI, plus other works in progress)
- Other System Performance Simulators (SPS)
  - Parametrical analysis
  - Statistical analysis
- Ground Processor Prototypes (GPP)
- Scene generators

7. Conclusions

- openSF represents a **generic software simulation infraestructure** that can be **easily configured and adapted** to any space mission with **minimal impact** on scientific models participating in the simulation.
- Core components are already produced and extra functionality is a continuous on-going process.

Thanks for your attention!

openSF

A Generic Framework for End-To-End Mission Performance Simulations