BIBLOS: BUILDING BLOCKS FOR EARTH OBSERVATION MISSION PERFORMANCE SIMULATORS

L. Soto[1], K. Szczepankiewicz[1], R. Kędzierawski[1], C. de Negueruela[2], A. Camps[3], H. Park[3], H. Metselaar[4], R. Franco[4]

GMV[1]
ul. Hrubieszowska 2, 01-209 Warsaw, Poland
E-mail: lsoto@gmv.com, kszczepankiewicz@gmv.com, rkedzierawski@gmv.com

GMV[2]
Calle Isaac Newton 11,Tres Cantos (Madrid), 28760 Spain
E-mail: cnegueruela@gmv.com

Universitat Politècnica de Catalunya [3]
C/Jordi Girona, 1-3 Barcelona, Spain
E-mail: camps@tsc.upc.edu, park.hyuk@tsc.upc.edu

ESA / ESTEC[4]
Keplerlaan 1, 2201 AZ Noordwijk, The Netherlands
E-mail: Harold.Metselaar@esa.int, Raffaella.Franco@esa.int

ABSTRACT

End-to-end mission performance simulators for Earth Observation missions are one of the prominent tools for system design and scientific validation in early mission phases. The European Space Agency (ESA) has promoted efforts to exploit synergies between activities and reduce engineering costs. Some of these activities are the ARCHEO study, the BIBLOS project, the OpenSF framework and the EO CFI library, the. The main goal of BIBLOS is to provide a library of software units called "Building Blocks", or simply "Blocks", that can be used to build an end-to-end simulator. Many Blocks are common across simulators, for example the geometry-related ones. Some Blocks are common for a certain type of instrument, like the Radiative Transfer Model, or parts of the Instrument model. BIBLOS targets the Blocks most frequently used by the engineering and scientific community. The user can access the library through the BIBLOS website, download the Blocks and use them directly, in combination with their own developments, or modify them. All of the Blocks are provided with the source code, and are under ESA Software Community License.

The first stage of this activity focused on Passive Optical instruments, mainly imagers, which are one of the most frequent types of instrument on Earth Observation satellites. The models already developed include the geometry, scene generation and instrument modelling of an optical imager. These Blocks can be combined into a full chain that produces raw data. ESA is currently developing the Level-1 processing, which may be included into BIBLOS in the future. The models, documentation and demos can be downloaded from the BIBLOS website: https://gmv-biblos.gmv.com/.

A second stage of the activity is currently ongoing with the purpose of expanding this library to include Passive Microwave instruments and Active Microwave instruments. Several simulators for Passive and Active Microwave payloads are currently being developed for ESA missions and it is foreseen that more will follow in the near future. Therefore, BIBLOS has the potential of supporting these developments.

Additionally, as part of a continuous improvement process, this second stage of BIBLOS will also update some of the most computational performance intensive blocks for Passive Optical instruments with parallel implementation for Graphic Processing Units.

This paper presents the work carried out for the second stage of the BIBLOS activity.

INTRODUCTION

The European Space Agency (ESA) has run several activities in order to reduce the re-engineering effort, promote reuse and standardisation in the development of end-to-end performance simulators (E2ES) for Earth Observation (EO) missions, [1]. Some of these activities are:

- **OpenSF:** open source simulation framework designed to support the development of E2ES, [2].
- **ARCHEO:** definition of a Reference Architecture for earth science E2ES, [3].
- **EO-CFI:** library of software functions performing accurate computations of mission related parameters for Earth Observation missions, [4][5].
- **BIBLOS:** The development of generic software models that can be used across missions, e.g. geometry, forward models, instruments models, [6][7].

In the last edition of Simulation for European Space Programmes (SESP) in 2015, GMV presented the work done in the BIBLOS project. In the first phase of the activity, BIBLOS developed software models for the instrument of a passive optical instrument (an imager or radiometer), [6]. A website was created where users can download and use the models. The project was well received and the second phase was launched. In this edition of the SESP, GMV will present the following developments of the BIBLOS project.

For BIBLOS-2, the library of software models is being extended in three areas:

- For passive optical instruments, the instrument modelling is being enhanced. Additionally, parallel programming algorithms have been developed for those models that are a bottleneck.
- One of the most relevant types of instruments for Earth Observation is Synthetic Aperture Radars (SARs), as was analysed during ARCHEO [3]. There are several future Earth Observation missions that will carry SAR instruments, [8]. BIBLOS-2 will provide models for mono-, bi- and multi- static SAR configurations, focusing on the Geometry and performance evaluation.
- Passive Microwaves are another relevant instrument for many applications in the Earth Observation (EO) domain. BIBLOS-2 will define and implement software models for the Geometry, Scene Generation, Instrument and Level-1 processing of Passive Microwaves.

BIBLOS DESCRIPTION

BIBLOS stands for **B**u**i**lding **Blo**cks for earth observation mission performance **S**imulators. It is a collection of software models that can be used in E2ES for EO missions. BIBLOS helps the user define the architecture, and provides validated units of software to help the user build its E2ES at a lower cost. In this section, first the Reference Architecture is presented, the concept that allows for standardisation and flexibility, as described in [1]. Secondly, the user guide for BIBLOS is outlined.

The library of software models for BIBLOS is composed of high-level Modules, that themselves are constituted of low-level Building Blocks.

The Reference Architecture

The simulation chain of any E2ES for EO missions can be divided into six high-level Modules: Geometry, Scene Generator, Instrument, Level-1 Processing, Level-2 Retrieval and Performance Evaluation. Fig. 1 shows the typical generic data flow that is the Reference Architecture. The Reference Architecture can be adapted for different mission types, for example, several instruments on board, formation flying, etc. For more information on the Reference Architecture and examples of other configurations please refer to [3] and [9].

The high-level Modules are:

- **Geometry Module.** In charge of simulating the spacecraft orbit and attitude, as well as the generation of the observation geometry of each instrument.
- **Scene Generator Module.** In charge of simulating the scene to be observed (land, ocean or atmosphere) and all environmental effects (radiative transfer models, atmosphere simulation, illumination conditions...) to be considered for the correct generation of the stimuli to be entered to the Instrument Module.

- **Instrument Module.** In charge of simulating the sensor behaviour, having different outputs depending on the type of instrument.
- **Level-1 Processing Module.** In charge of the generation of level-1 products, from level-1a to level-1c.
- **Level-2 Retrieval Module.** In charge of performing the retrieval of the geophysical parameters that are the objective of the mission/instrument. Depending on the mission and on its definition of the products, this Module would generate level-2 data or products at a higher level of processing.
- **Performance Evaluation Module.** In charge of performing the needed analysis of the simulator outputs to evaluate the performances of the mission. It could be run at different points of the simulation chain.

Each of these high-level Modules is composed of Building Blocks. A Block is a unit of software that performs certain functionality.

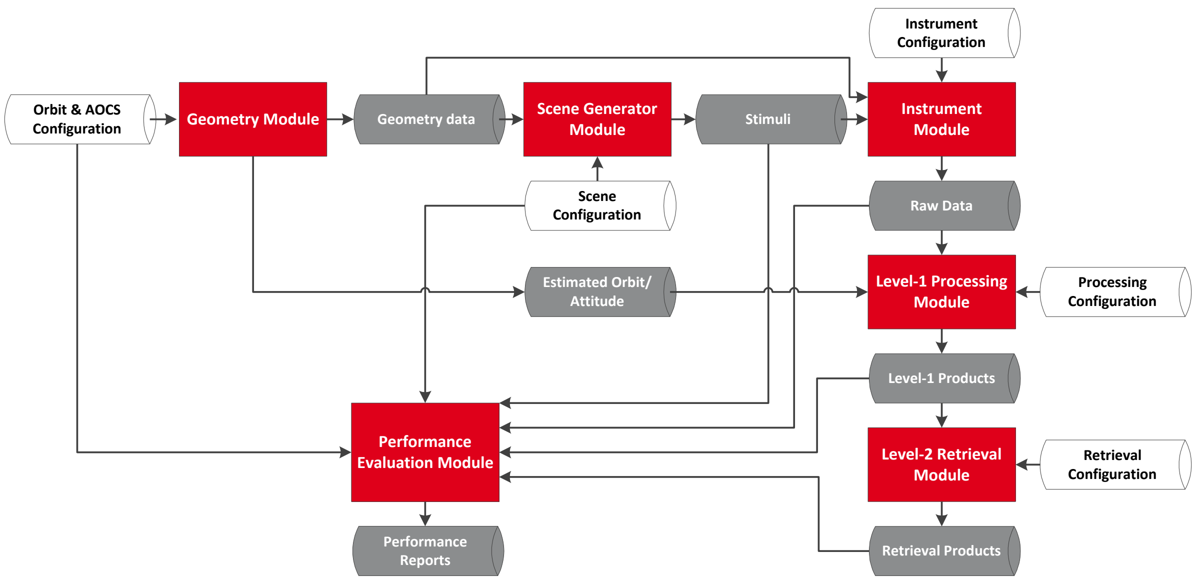

Fig. 1. Generic data flow at the highest level of the Reference Architecture

BIBLOS-2 SCOPE

The details on the scope of the second phase of the BIBLOS project are detailed in this section, divided by instrument type.

Active Microwaves

A full Synthetic Aperture Radar end-to-end simulator is a very complex development. For BIBLOS-2, following the interest expressed by ESA, it has been decided to focus the development on the Geometry Module to support multistatic configurations. This way, BIBLOS will be immediately useful for upcoming ESA missions. In a monostatic radar the transmitter and receiver antenna are located in the same position. In recent years there has been an increase in the interest of bi- and multistatic configurations, where the transmitter and receiver are located in different platforms, [10].

Additionally, BIBLOS-2 will include a series of evaluation functionalities including: Diamond diagrams, 2D resolution plots, Noise-Equivalent Sigma Zero, Ambiguity ratios, cross-sections of the antenna pattern, footprints, Impulse Response Function, and swath geometry.

Passive Microwaves

The scope of the Passive Microwaves activity is to model a typical conical scanning microwave radiometer (see Fig. 2) that takes images in the microwave or millimetre wave part of the spectrum at constant incidence angle, with spatial, spectral and radiometric resolutions varying for the mission. On one side, the Passive Microwaves models will implement dual polarization brightness temperatures of the Earth's surface and radiative transfer models of the atmosphere so as to generate the brightness temperatures top of the atmosphere (TOA) in the antenna reference frame. On the other side, it will implement the receivers' chain model, including thermal drifts, a noise model as a function of the integration time (i.e. Allan's variance model), and a digital back-end with a number of radio-frequency interference detection and mitigation techniques. Without loss of generality, the full instrument and the Level-1 processing chain will be tailored for a mission based on EUMETSAT Polar System-Second Generation Microwave Imager instrument (MWI).

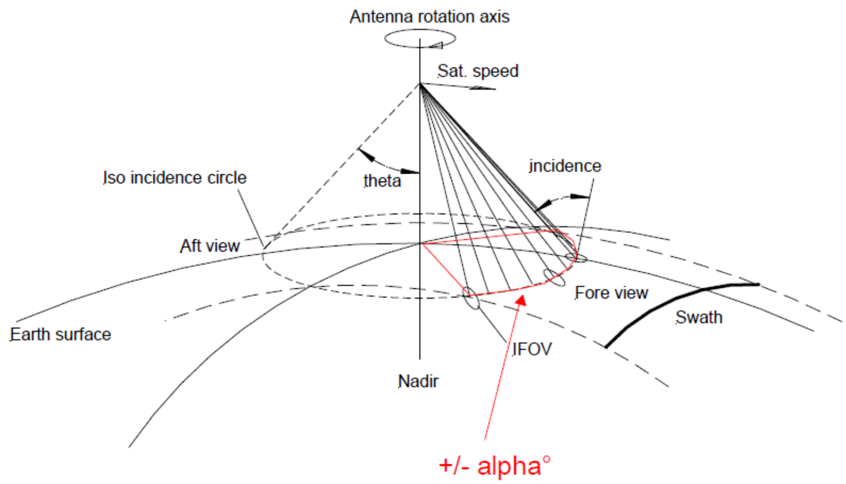

Fig. 2. Scan geometry of the MWI instrument. Adapted from EUMETSAT through [11]

Passive Opticals

For the first phase of BIBLOS the Geometry, Scene Generation and Instrument Modules for an imager instrument have been implemented. For this extension, the following features shall be included:

- Include aspects for a finer spectral modelling by applying (via configuration) spectral dependent parameters before the band merging, at the beginning of the Instrument Module.
- Add effects to the Instrument Module like optical (diffraction, aberrations, defocus, realisation), detector, smearing, motion blur, desynchronization and binning. The option to input an externa MTF for other effects will be kept.
- Include non-linearity effect.

Another area of work, described in detail in the following section, is the improvement of the performances of the blocks identified as bottlenecks.

COMPUTATIONAL PERFORMANCE

Performance is one of the concerns that is taken into consideration during the second phase of project development. Since some of the processing algorithms' demands for computational power are high, it is typical to refer to parallel computing in order to improve performance. This section presents the BIBLOS performances and Hardware/Software solutions that were analysed in the context of BIBLOS 2 activities.

Parallel computing

The hardware solutions for modern PCs that support parallel processing are:

- **Graphic Processing Units (GPU).**
- **Multi-core Central Processing Units (CPUs).** Together with Hyper-Threading which logically doubles the number of cores, multi-core CPUs are a powerfull technology that facilitates data processing parallelisation.

GPUs differ from Central Processing Units (CPUs) in more transistors dedicated to data processing than controlling activities, which is presented in Fig. 3. Different architecture of GPUs results in specific programming model and dedicated language extensions (available for C, Fortran, Java, C# and others). GPUs introduce a heterogeneous programming model based on kernels, hierarchical memory (global, shared and local) and thread hierarchy that organizes parallel threads into 3-dimensional blocks in 3-dimensional grid. The introduction of another programming model adds more complexity to the software architecture and development, but the usage of many thousand psysical threads in graphic cards results in achieving better performance in applications, where large datasets can be processed simultaneously and independently.

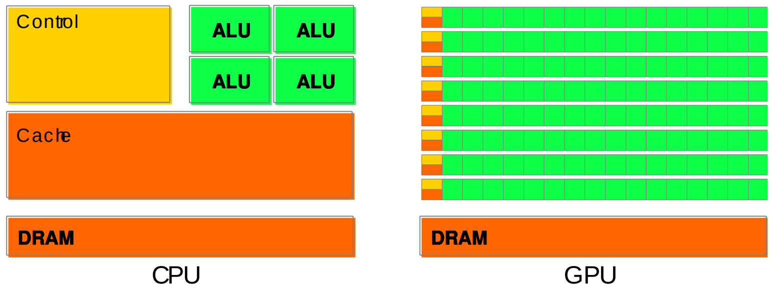

Fig. 3. The GPU Devotes More Transistors to Data Processing, based on Fig. 3 in [13].

Multi-core CPUs together with Hyper-Threading technology create an environment that consists of about 8-32 logical cores which can be used in parallel execution and data processing. The most known application programming interface that supports the creation of applications that make use of many CPU cores for parallel data processing is OpenMP [12]. OpenMP provides scalable programming model and flexible interface for programmers. It implements multithreading, a model where one master thread creates its slave threads to split tasks among them. Communication is realized via shared memory. Together with Message Passing Interface, OpenMP can be used to create multithreaded application that is able to run on several physical machines. It is also worth mentioning that C++ standard (since C++11) implements parallel version of algorithms using OpenMP.

The most popular technology that supports General-Purpose Computing on Graphic Processing Units (GPGPU) is Compute Unified Device Architecture (CUDA) [13] released by NVIDIA [14]. The second place belongs to Open Computing Language (OpenCL) [15] that is API for heterogeneous computing, running on CUDA architecture. OpenCL supports both top producers of graphic cards: NVIDIA and AMD [16]. Although designed for one hardware provider, CUDA provides many optimized libraries dedicated for example to signal processing (cuFFT), algebraic operations (cuBLAS), solving equations (cuSolver), random data generation (cuRAND), graph analytics library (nvGRAPH) or neural networks (cuDNN) [17]. A comparison table with advantages and disadvantages of each technology is shown in Table 1.

Table 1. Comparison of technologies supporting parallel processing.

Technology	Advantages	Disadvantages
CUDA	**Best performance**. Application could be several hundred times faster than with CPU processing. Best performance shortens simulation from several days/weeks to several hours. Preliminary estimation of mission simulation for 50-100km scene would take weeks without parallelisation. **Support**. Optimized libraries exist for several purposes, e.g. cuFFT, cuRAND. This reduces implementation effort and improves maintenance and evolution. CUDA does not exclude other parallelization methods which can shorten the time of execution.	**Implementation effort**. Much more effort required to implement application. Application written in CUDA cannot run on CPU without additional coding process. **Compatibility.** It requires NVIDIA graphic card. These cards are not very expensive, a basic one can cost around 200 Eur. This is important for Universities, and small institutions. Can be a problem in MAC computers – no NVIDIA cards for Apple products.
OpenCL	**Performance**. Shortening of simulation time and testing process. Slightly worse performance than CUDA. **Compatibility**. Most GPUs Can run on PC without CUDA and on AMD graphic cards.	**Implementation effort.** 10-20% more than CUDA effort added to implementation time. More API devoted to device/program handling. Available runtime compiler. More effort added to testing.

		Support. Leading technology for GPU computations is still NVIDIA CUDA. There is not much additional libraries. This is a key factor during phase of implementation, code maintenance and evolution.
OpenMP	**Performance*.** Advantage when using in applications that process less data. **Compatibility**. Can be used for code parallelization without additional GPU card. **Implementation effort*.** Advantage when using parallel extensions from standard library.	**Implementation effort*.** Disadvantage when using OpenMP API. Around 10% less effort added to implementation time in comparison to CUDA) **Performance*.** When better data throughput is required, performance is remarkably worse (even hunded times in some cases) than for GPU processing.

The technology that seems to be the best for code parallelization is CUDA. It outperforms OpenMP and is slightly better than OpenCL, taking into account computational power. CUDA has the best support; many libraries were developed by NVIDIA and other communities. However, one of the crucial disadvantages is that it requires an NVIDIA graphic card to perform calculations. While it is not a problem for a modern PC, it is a disadvantage for the all Apple MAC products, since Apple stopped releasing MAC computers with NVIDIA graphic cards. Taking into consideration that many users, including personnel from ESA, work using MAC products, OpenCL is the best option to be chosen for BIBLOS computational performance optimisation.

BIBLOS computational performance

BIBLOS performances were measured in one of the first activities of BIBLOS 2 project. The measurements are enclosed in Table 2. Calculations were performed using the following hardware and software:

- Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz (8 cores) 64 bit
- 8GB RAM memory
- OS: Linux Opensuse 42.1 64 bit, kernel 4.6.4-5.g8f4696b-default
- EOCFI 4.9 (see [5])
- OpenSF 3.3 with OSFI

Table 2. Comparison of technologies supporting parallel processing.

Block name	Execution time (20km orbit)
Orbit Block	~30 seconds
Attitude Block	~30 seconds
AOCS/Instrument Coupling Block	~60 seconds
Scene Interaction Block	~36 hours
Resampling Block	~1 hour
Atmosphere Simulator Block	~2 hours
Spatial Block	~360 seconds
Radiometric Block	~13 seconds

The blocks that need to be parallelised within BIBLOS 2 activities are:

- Scene Interaction Block
- Resampling Block
- Atmosphere Simulator Block

BIBLOS WEBSITE

User interface

BIBLOS website (see [7]) created in the first phase of BIBLOS development allows registered users to access and download the BIBLOS code, documentation and test data. The current interface of the BIBLOS website is shown in Fig. 4.

Content Organisation

The hierarchy of the website content is: Instrument > Module > Block. The user can select the type of instrument to view the high-level Modules. Then a user can select the specific block and download the package with the following content:

- Source code. All the code needed to compile BIBLOS block.
- Block documentation.
- User manual.

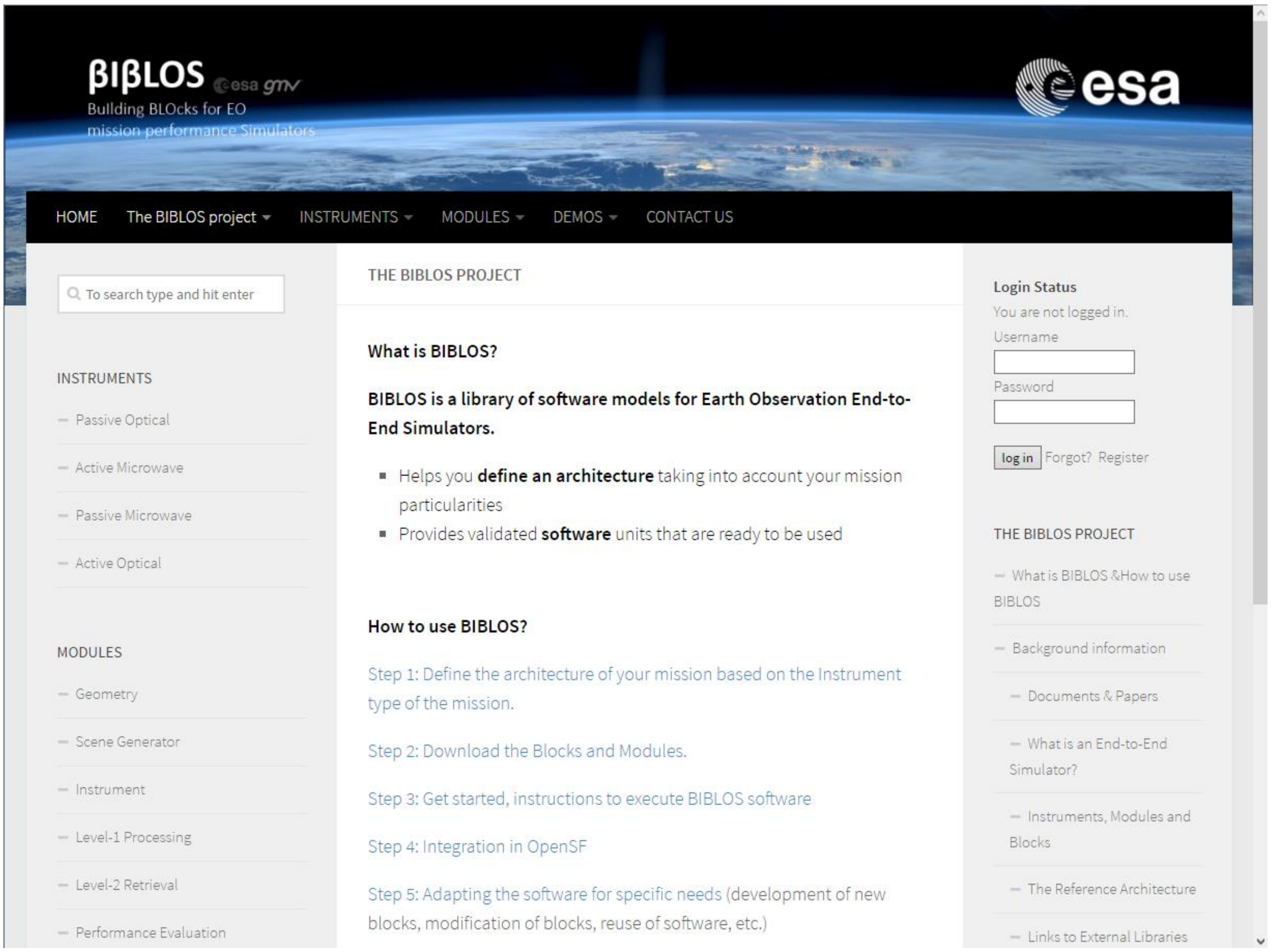

Fig. 4. The BIBLOS website [7].

All the Passive Optical Instrument simulator blocks can be accessed and downloaded at once via item: “Multi-band Imager – Instrument data simulator, Proof of Concept”.

How to use BIBLOS

BIBLOS is designed to make the developer of an E2ES save engineering effort. Through the website, the user is guided on how to build its simulator in 5 basic steps:

Step 1: Define the architecture of the mission based on the Instrument type of the mission.
Step 2: Download the Blocks and Modules.
Step 3: Get started, instructions to execute BIBLOS software
Step 4: Integration in OpenSF

Step 5: Adapting the software for specific needs (development of new blocks, modification of blocks, reuse of software, etc.)

BIBLOS license

BIBLOS is under ESA Software Community License. Details can be found in the website, [18].
The website of BIBLOS is currently hosted by GMV, during the development phase. In the future it will be hosted by ESA's Earth Observation Programmes portal.

CONCLUSIONS

The BIBLOS project is a practical guide for users who want to develop an Earth Observation performance simulator. This second phase has an ambitious scope that includes the three most relevant types of instrument in the Earth Observation domain, [3]: Active Microwaves, Passive Microwaves and Passive Opticals.
BIBLOS already has available for the user community models for the Passive Opticals, and as this activity progresses, it will include models for the other two types of instruments.
This activity will have an impact in the saving of reengineering effort in future EO E2ES.

REFERENCES

[1] R. Franco and J. Fuchs, *ESA Mission Scientific Validation By Means Of End-To-End Performance Simulators* (2016), IAC-16, D1.3.4

[2] R. Moyano and E. del Pozo, *OpenSF: A Generic Framework For End-To-End Mission Performance Simulations* (2010), Simulation for European Space Programmes.

[3] C. de Negueruela, et al., "ARCHEO-E2E: A Reference Architecture for Earth Observation end-to-end Mission Performance Simulators", SESP 2012

[4] ESA, *Model Library for Earth Observation End-to-End Simulators Statement of Work* (2014), TEC-SWM/14-696/RF.

[5] ESA. *EOCFI* [online] http://eop-cfi.esa.int/mission-cfi-software/eocfi-software

[6] Lucia Soto, Cristina de Negueruela, Artur Kobyłkiewicz, Agata Orych, Raffaella Franco. *EOMODEL: A model library for Earth Observation End-to-End Simulators.*

[7] GMV, *BIBLOS* [online] https://gmv-biblos.gmv.com

[8] ESA. *ESA Future Missions*. Accessed February 28, 2017. https://earth.esa.int/web/guest/missions/esa-future-missions.

[9] GMV, *ARCHEO-E2E EO E2E Reference Architecture* (2013), GMV-ARCHEO-E2E-TN-002.

[10] Bezoušek, Pavel, and Vladimír Schejbal. "Bistatic and Multistatic Radar Systems." *Radioengineering* 17, no. 3 (September 2008). http://www.radioeng.cz/fulltexts/2008/08_03_053_059.pdf.

[11] *MetOp-SG (MetOp-Second Generation Program).* EOportal. Accessed February 28, 2017. https://directory.eoportal.org/web/eoportal/satellite-missions/m/metop-sg.

[12] *The open standard for parallel programming of heterogeneous systems* [online] https://www.khronos.org/opencl/

[13] *CUDA C Programming Guide* [online] http://docs.nvidia.com/cuda/cuda-c-programming-guide

[14] *NVIDIA website* [online] http://www.nvidia.co.uk/

[15] *The open standard for parallel programming of heterogeneous systems* [online] https://www.khronos.org/opencl/

[16] *AMD website* [online] http://www.amd.com/en-us

[17] *GPU-Accelerated Libraries* [online] https://developer.nvidia.com/gpu-accelerated-libraries

[18] *BIBLOS Software License.* ESA through GMV. Accessed February 28, 2017. https://gmv-biblos.gmv.com/biblos-software-license/.