Increasing Performance of ESA Operational Spacecraft Simulators

September 2012

V. Reggestad, M. Pantoquilho, D. Werner
ESA/ESOC, HSO-GD, Mission Data Systems Division

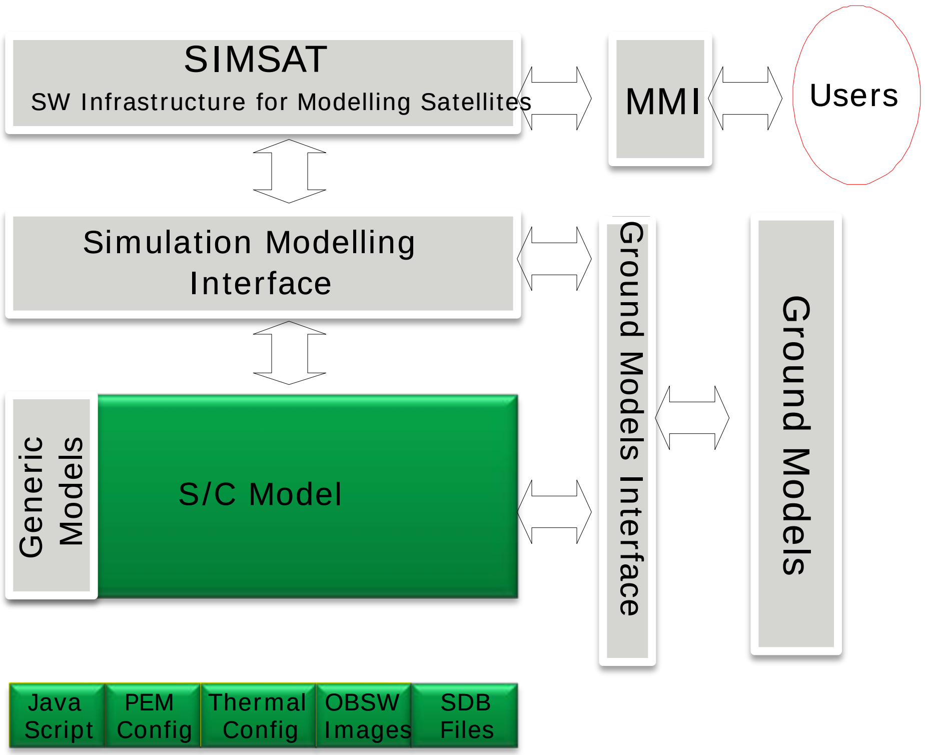
N. A.I.Livanos, P. Antoniou
EMTech, Greece

AGENDA

- Operational Spacecraft Simulators
- Performance Indicator Tools (the PIT)
- Approach to Code Performance Optimizations
- Performance Analysis: Some Results
- Conclusions & Future Work

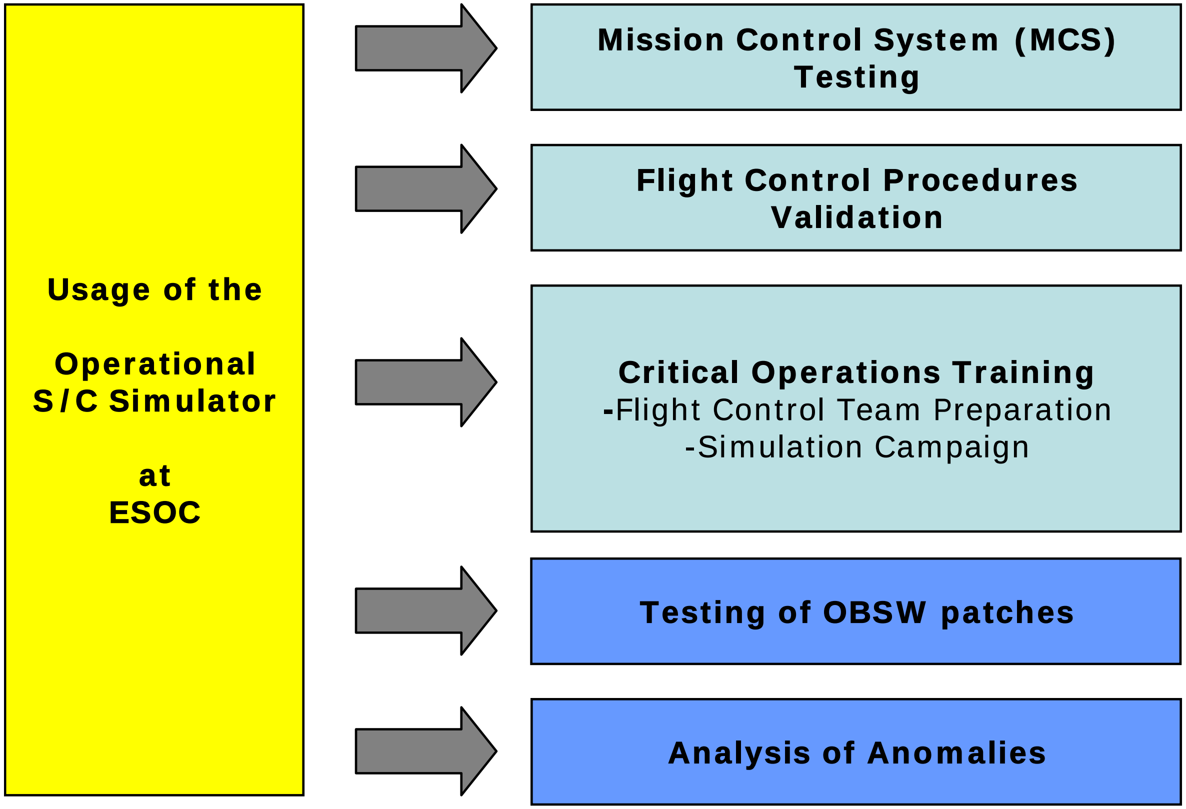
OPERATIONAL S/C SIMULATORS

- Complex piece of software
- Mimics real S/C
- Runs the real S/C OBSW
- Accepts the real TCs
- Produces realistic TM
- Soft real time + faster then real time
- Developed for a specific mission
- Based on reusable infrastructure software (SIMULUS)



European Space Agency

OPERATIONAL S/C SIMULATORS USAGE



Usage of the

Operational S/C Simulator

at ESOC

Mission Control System (MCS) Testing

Flight Control Procedures Validation

Critical Operations Training
-Flight Control Team Preparation
-Simulation Campaign

Testing of OBSW patches

Analysis of Anomalies

Operational S/C Simulator's Lifetime = Mission's Lifetime

SIMULUS

- SIMSAT

  (Run-time framework plus set of generic MMIs)

- Software emulators

  (running the actual on-Board Software)

- Generic Models

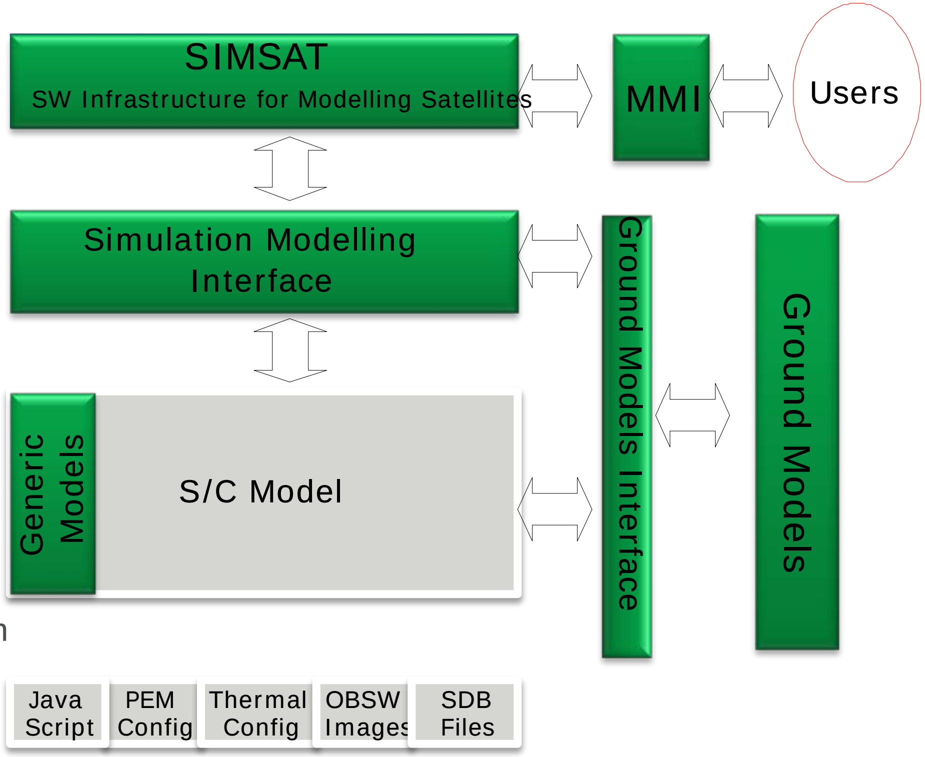
- Ground Models

  (Station equipment models).

- SMP2 standard compliant

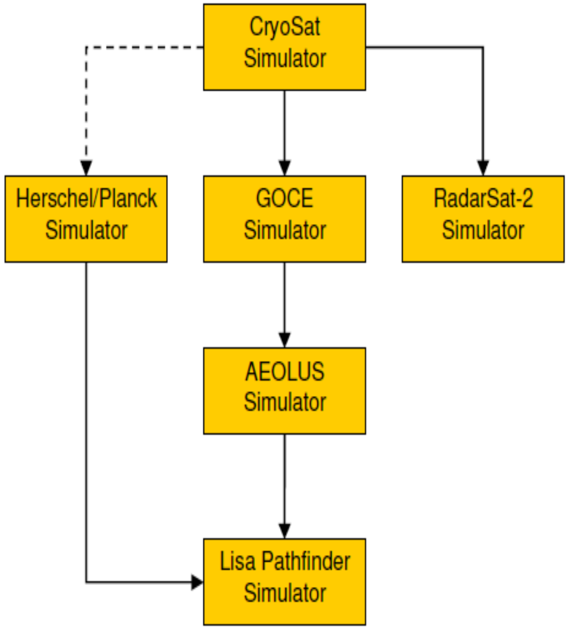
  (standardized interfaces between the simulation models and between them and the simulation run-time environment)

- REFA: Reference Architecture (reference operational S/C simulator architecture)



CODE RE-USE: BENEFITS & DRAWBACKS

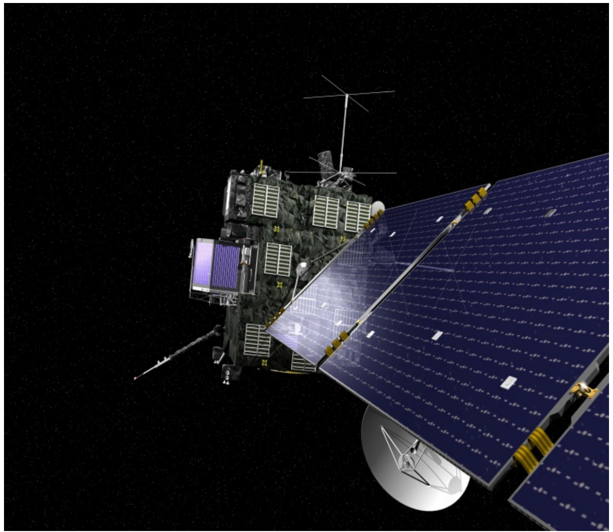
- Strong incentive to re-use code among simulators (for common subsystems)
- Reduces cost and implementation time
- Increases the life time of a simulator's code far beyond the mission's lifetime:
  - Un-optimized code may have a life span of several missions
- Results in:
  - Increased need to produce highly performing code;
  - And/or to optimize reused or reusable code for the benefit of current and future missions.



Operational S/C Simulator's **Code** Lifetime **>** Mission's Lifetime

Optimization of SIMULUS Simulators

- Why ?
- How to analyse (and improve) performance of SIMULUS based Simulators ?



European Space Agency

AGENDA

- Operational Spacecraft Simulators
- Performance Indicator Tools (the PIT)
- Approach to Code Performance Optimizations
- Performance Analysis: Some Results
- Conclusions & Future Work

Performance Indicator Tools (PIT)

- support real-time metric measurements;
- provide performance evaluation of sub-systems and parts of the overall software;
- designed and implemented as a separate shared library;
- interfaced from inside the tree in the SIMSAT environment through the configuration of an XML file;

- provide the opportunity to build a performance perspective of a S/C operational simulator (including mission specific and SIMULUS software)

- The PIT are subdivided into 4 categories and a presentation tool:

- **E**vent **R**eporting **S**ystem
- **E**xternal **P**rofiling **T**ools
- **I**nternal **P**erformance **M**etrics
- **S**IMSAT **P**erformance **M**etrics

- **O**pen **O**ffice **M**acro **L**ibrary

Performance Indicator Tools (contd')

PIT-ERS: Event Reporting System

(generation of event & function calls)

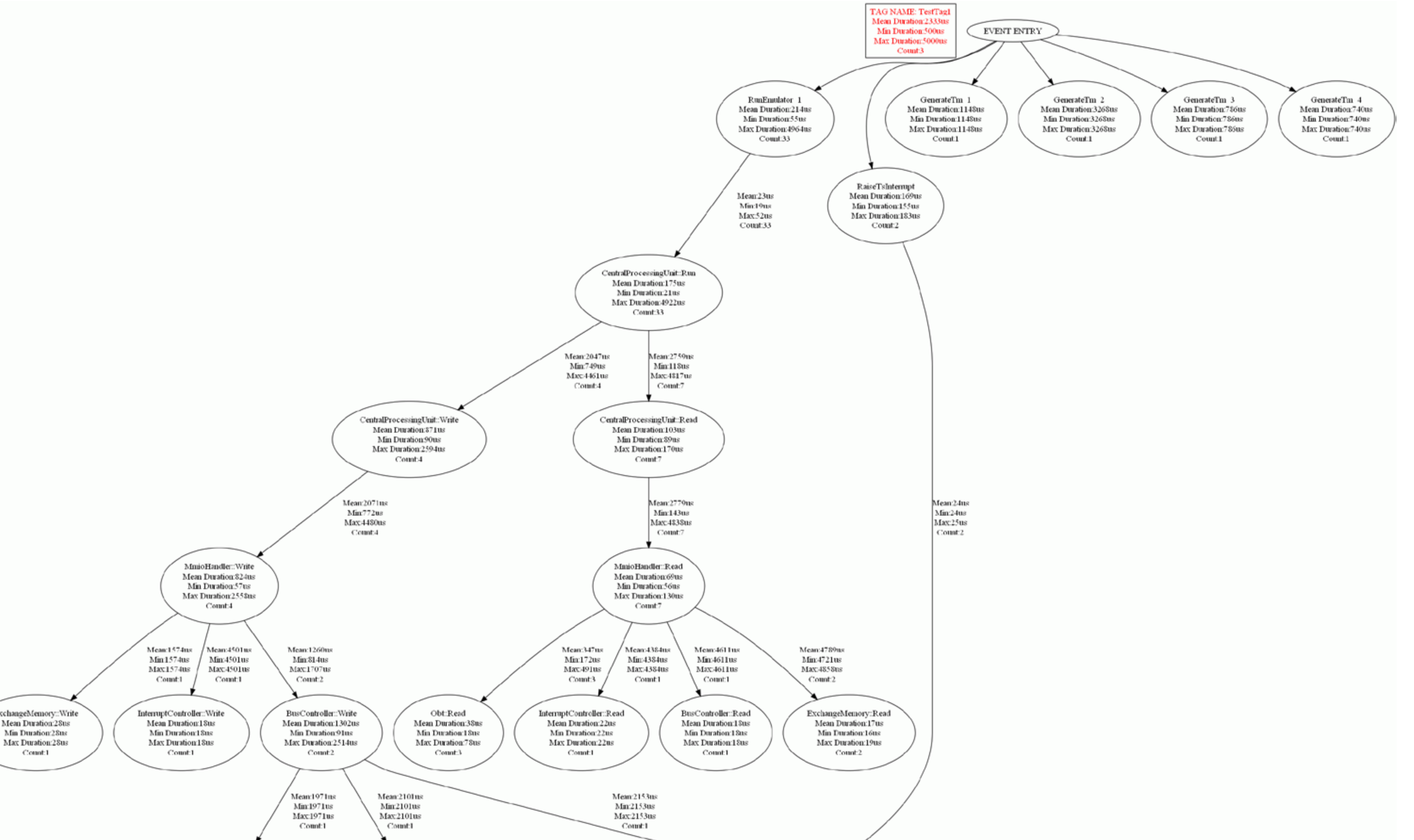
- acquires information of events and function calls generated in SIMSAT kernel by the execution of SMP2 simulation models;
- tracks all the function calls related with the activated event;
- generates analytical call graphs of the simulation system;
- outputs are compatible with the GraphViz tool.
- analysis of the graphs can be used to manually explore parallelization.

PIT-EPT: External Profiling Tools

(interfacing to external profiling tools)

- uses O-Profile as 3rd Party External Profiling Tool.
- provides a fast and easy way to perform CPU profiling of all software components.
- performs accumulations of the generated information according to user defined categories.
- continuously updates the results inside the PIT-ERS tree structure in the SIMSAT environment.

Event Reporting Output



TAG NAME: TestTag1
Mean Duration:2333us
Min Duration:500us
Max Duration:5000us
Count:3

EVENT ENTRY

RunEmulator 1
Mean Duration:214us
Min Duration:55us
Max Duration:4964us
Count:33

GenerateTm 1
Mean Duration:1148us
Min Duration:1148us
Max Duration:1148us
Count:1

GenerateTm 2
Mean Duration:3268us
Min Duration:3268us
Max Duration:3268us
Count:1

GenerateTm 3
Mean Duration:786us
Min Duration:786us
Max Duration:786us
Count:1

GenerateTm 4
Mean Duration:740us
Min Duration:740us
Max Duration:740us
Count:1

Mean:23us
Min:19us
Max:52us
Count:33

RaiseTsInterrupt
Mean Duration:169us
Min Duration:155us
Max Duration:183us
Count:2

CentralProcessingUnit::Run
Mean Duration:175us
Min Duration:21us
Max Duration:4922us
Count:33

Mean:2047us
Min:749us
Max:4461us
Count:4

Mean:2759us
Min:118us
Max:4817us
Count:7

CentralProcessingUnit::Write
Mean Duration:871us
Min Duration:90us
Max Duration:2594us
Count:4

CentralProcessingUnit::Read
Mean Duration:103us
Min Duration:89us
Max Duration:170us
Count:7

Mean:2071us
Min:772us
Max:4480us
Count:4

Mean:2779us
Min:143us
Max:4838us
Count:7

Mean:24us
Min:24us
Max:25us
Count:2

MmioHandler::Write
Mean Duration:824us
Min Duration:57us
Max Duration:2558us
Count:4

MmioHandler::Read
Mean Duration:69us
Min Duration:56us
Max Duration:130us
Count:7

Mean:1574us
Min:1574us
Max:1574us
Count:1

Mean:4501us
Min:4501us
Max:4501us
Count:1

Mean:1260us
Min:814us
Max:1707us
Count:2

Mean:347us
Min:172us
Max:491us
Count:3

Mean:4384us
Min:4384us
Max:4384us
Count:1

Mean:4611us
Min:4611us
Max:4611us
Count:1

Mean:4789us
Min:4721us
Max:4858us
Count:2

xchangeMemory::Write
Mean Duration:28us
Min Duration:28us
Max Duration:28us
Count:1

InterruptController::Write
Mean Duration:18us
Min Duration:18us
Max Duration:18us
Count:1

BusController::Write
Mean Duration:1302us
Min Duration:91us
Max Duration:2514us
Count:2

Obt::Read
Mean Duration:38us
Min Duration:18us
Max Duration:78us
Count:3

InterruptController::Read
Mean Duration:22us
Min Duration:22us
Max Duration:22us
Count:1

BusController::Read
Mean Duration:18us
Min Duration:18us
Max Duration:18us
Count:1

ExchangeMemory::Read
Mean Duration:17us
Min Duration:16us
Max Duration:19us
Count:2

Mean:1971us
Min:1971us
Max:1971us
Count:1

Mean:2101us
Min:2101us
Max:2101us
Count:1

Mean:2153us
Min:2153us
Max:2153us
Count:1

Performance Indicator Tools (contd')

PIT-IPM – Internal Performance Metrics

(measurements inside S/C models)

- benchmarks performance of subsystems (e.g. REFA, GENM, S/C models);
- measures the respective total throughput in specific subsystem utilization of the S/C (e.g. nr. of Tele-commands and Telemetry packets per second, nr. of OBSW memory inputs & outputs, nr. of bus transactions);
- used to develop stress tests of S/C subsystems.

PIT-SPM - SIMSAT Performance Metrics

(measurements inside the Simulation Kernel)

- includes performance metrics and functionalities related to SIMSAT environment (i.e. scheduler, kernel, object access, script code execution, etc.).
- measurements are rates and counters.

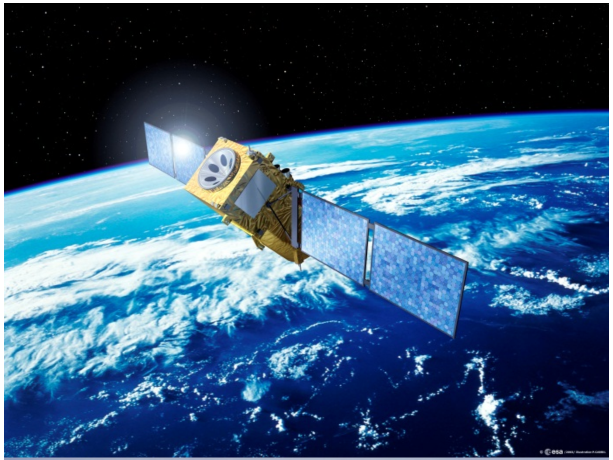
PIT-OOML - Open Office Macro Library

Performance Analysis: Preliminary Results

- Biggest performance problem is the emulator core:
  - it consumes most of the CPU time;
  - observed in Cryosat2, Aeolus, Hershel&Planck Operational Spacecraft Simulators.
- Further exploitation of Parallelization is possible.

Performance Optimization of SIMULUS based Simulators

- Performance optimization for SIMULUS infrastructure and SIMULUS based Operational S/C Simulators must consider:
  - Emulator optimization & enhancement;
  - Parallelization;
  - **Code optimization:**
    - of SIMULUS software ;
    - and S/C models software.

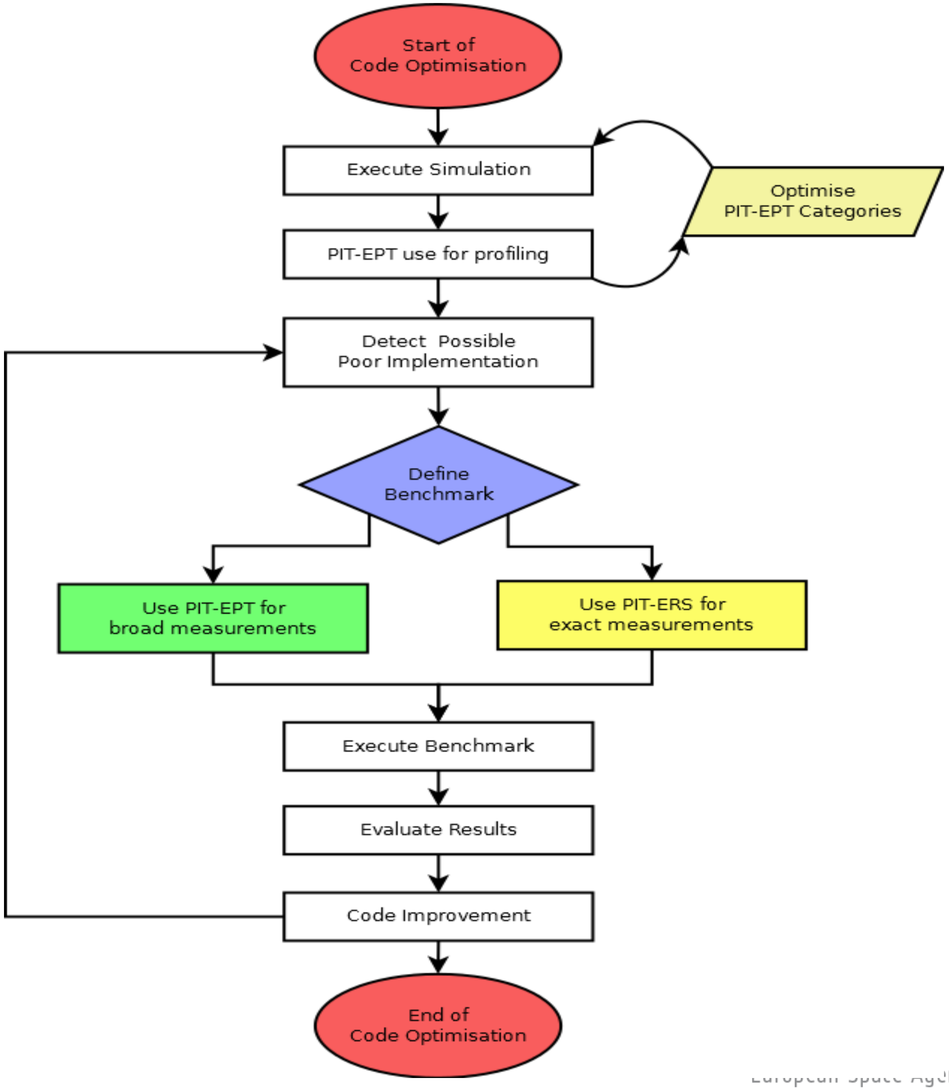


AGENDA

- Operational Spacecraft Simulators
- Performance Indicator Tools (the PIT)
- Approach to Code Performance Optimizations
- Performance Analysis: Some Results
- Conclusions & Future Work

The PIT Code Optimization Approach

- Applicable:
  - During Development (by the developers)
  - During Validation (by the costumer)



AGENDA

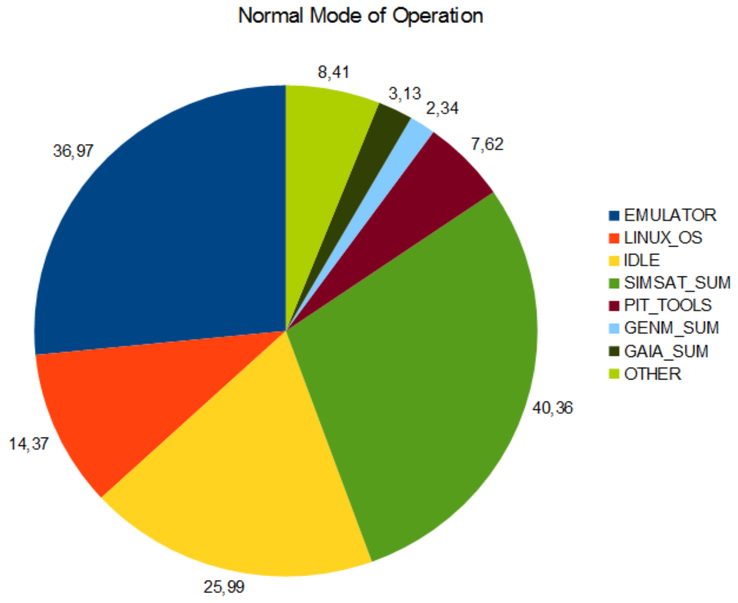
- Operational Spacecraft Simulators
- Performance Indicator Tools (the PIT)
- Approach to Code Performance Optimizations
- Performance Analysis: Some Results
- Conclusions & Future Work

Preliminary Results

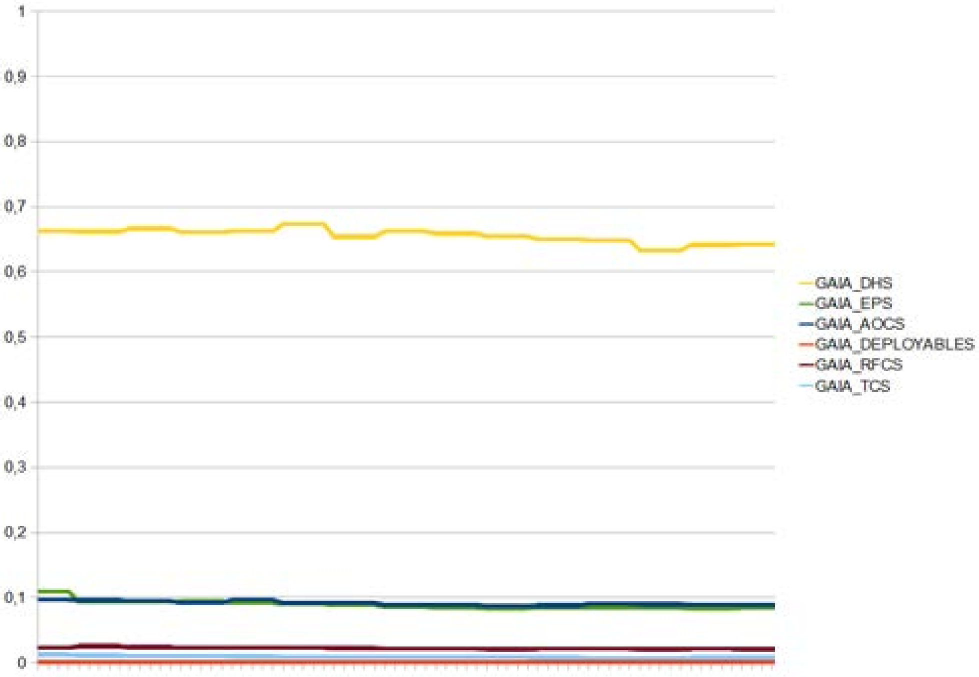
- Biggest performance problem with highest CPU time consumption in all analysed simulators is the emulator core.
- Cryosat2 analysis:
  - high occurrence of function call invocations of non-local functions (located in other shared objects).
  - high occurrence of string compare functions.
- Aeolus analysis:
  - thermal network simulation takes most of the CPU time - $O(n^2)$ implementation.
  - high occurrence of string and memory copy functions, of big amounts of data.
- Herschel and Planck analysis:
  - on a low enough load the SIMULUS infrastructure functions and the TTC streams activity is quite high.

GAIA's Preliminary Results

- Load percentage of the used CPU for the whole GAIA system (including OS and SIMULUS) while running at the speed factor of 1x (real time).



- Used CPU percentage for several parts of the GAIA simulator (low load, with OBSW running).



European Space Agency

AGENDA

- Operational Spacecraft Simulators
- Performance Indicator Tools (the PIT)
- Approach to Code Performance Optimizations
- Performance Analysis: Some Results
- Conclusions & Future Work

CONCLUSIONS

- Preliminary results are promising;
- On-going project to industrialize the tools and to possibly distribute them with SIMULUS;
- GAIA could be the first simulator to test the re-engineered version of the PIT and to be able to benefit from the performance optimizations that will be suggested by the application of the tools;
- Future missions will be able to fully benefit from these tools and apply them early during their development phases.

PIT will be able to improve the performance and
code optimization of newly developed SIMULUS based simulators.

Thank you for your attention !

Any Questions ?

European Space Agency

Contacts:
Vemund.Reggestad<at>esa.int
Marta.Pantoquilho<at>esa.int