Introducing Parallelization & Performance Optimization in SIMULUS Based Operational Simulators

Vemund Reggestad
ESA/ESOC
Robert-Bosch-Str. 5
D-64293 Darmstadt, Germany
+496151902685
Vemund.reggestad@esa.int

Nikolaos-A.I.Livanos
EMTECH
Vizantiou 58 Str., Papagou
156 69, Athens, Greece
+302106528527
nikolaos.livanos@emtech.gr

Pantelis Antoniou
EMTECH
Vizantiou 58 Str., Papagou
156 69, Athens, Greece
+302106561059
panto@emtech.gr

Elias Zois
RDTL, Tech. & Educ. Instit. Of Athens
Ag.Spiridonos, Aegaleo
122 10, Athens, Greece
+302105385383
ezois@teiath.gr

ABSTRACT

As part of the ESA/ESOC study "Linux & Multi Core Processor Technology for Simulators" [7], several concepts concerning performance optimization of SIMULUS based operational simulators were introduced. Starting from the possibility to explore parallelism and distributed execution of computational loads (PDES approach) and continuing with the profiling, analysis and focused optimization of specific simulation models our efforts and ideas converged to a Performance Optimization Framework (POF). The present paper summarizes the way the idea of a Performance Optimization Framework can be applied in future operational simulators.

General Terms

Algorithms, Measurement, Documentation, Performance, Design, Experimentation.

Keywords

SIMULUS, ESA/ESOC Operational Simulators, PDES, Performance Optimization, Performance Indicator Tools (PIT)

1. INTRODUCTION

A principal objective of the present study was research and development on the required prerequisites for a high performance simulator able to run efficiently on multi-core processor technology. This included technological advancements in the development of algorithms for a scheduler which can take advantage of modern multi-core CPUs, as well as achievement of higher levels of performance and parallelism by improved model construction and annotation. Within the area of multi-core support, one important objective was to allow events that take place in parallel on real spacecrafts to also run in parallel within the operational simulators.

The study provided the opportunity to investigate the possibility of transforming the kernel simulation engine from a Discrete Event Simulation (DES) concept to a Parallel Discrete Event Simulation (PDES) one [12,16,17]. The attempt may be considered as a proof-of-concept for introducing parallelization into the Simsat simulation engine in respect with recent designs of spacecraft operational simulators.

The Performance Optimization Framework (POF) is a concept aiming to explore the utilization of the PDES simulation engine. It includes several tools and utilities, as well as manual and automatic procedures that focus on the profiling, analysis and preparation of performance optimized execution instances.

2. SIMULUS BASED OPERATIONAL SIMULATORS & PERFORMANCE ISSUES

2.1 SIMULUS Infrastructure

Satellite simulators at ESA/ESOC are built-up on top of the SIMULUS infrastructure [1], [2]. The latter comprises of a run-time framework (Simsat), software emulators and a set of reusable generic models. Recent Operational Simulator developments are also based on a Reference Architecture (REFA) [3], [4]. The latter has been introduced in order to identify, define, develop and integrate a reference spacecraft simulator architecture representing an initiative for further development. REFA and all future operational simulators that will be based on REFA are compliant with the SMP2 standard, [5], implementation of the models.

A key module inside an operational simulator is the On-Board Software (OBSW) emulation, [6]. The concept is to utilize the real mission's OBSW in the core simulation while other parts of the spacecraft are being modeled in a functional manner. All

Simutools 2012, March 19-23, Desenzano del Garda, Italy
Copyright © 2012 ICST 978-1-936968-47-3
DOI 10.4108/icst.simutools.2012.247832

simulation models, including the emulator, are executed by the Simsat simulation environment. The latter comprises of several modules providing all kernel functionalities required to support run-time simulation. The kernel includes a scheduling and execution engine that is based on a typical Discrete Event Simulation concept [11]. Events are generated inside the execution of simulation models and are stamped with specific simulation time. Events are linked with function callbacks that correspond to simulation models' functionalities. In turn, these may schedule new events and so on. The simulation's kernel scheduler serializes all event execution and keeps the timing of the overall execution using a global clock mechanism.

2.2 Targeting Performance Optimization

When considering performance optimization for SIMULUS infrastructure three are the major directions that a designer and a system integrator must take into account: parallelization, code optimization, and emulator optimization & enhancement. The first two directions were elaborated in the present study.

A major difficulty in any parallelization effort is to separate and group independent tasks, able to be allocated in a separate computation engine (e.g. a separate core in a multi-core processor) and in the same time provide a means of synchronization and communication (if required) between the distributed execution nodes. As far as a SIMULUS based operational simulator is concerned, the aforementioned inquiry drives to a first requirement. The Simsat simulation engine must be extended and support parallel execution of events, i.e. be enhanced with a Parallel Discrete Event Simulation (PDES) scheduling and execution engine [12].

In parallel discrete event simulation, the physical system is usually modeled as a collection of sequential simulations that represent different parts of the simulated system. These simulations are called "Logical Processes (LPs)". Just as in the sequential case, these LPs have their own state and process and generate events. Consequently, each logical process has its own local event list, local simulation clock and local state vector, fig. 1. PDES mechanisms broadly fall into two categories: conservative and optimistic. By applying conservative techniques, the possibility of events executing in an incorrect order is avoided in the first place. These approaches rely on some strategy to determine when it is safe to process an event. On the other hand, optimistic approaches use a detection and recovery approach: incorrect order of execution is detected and a rollback mechanism is used to return to a previous and safe state.

The modification of the SIMSAT kernel to support a naive PDES scheduler was a key activity of the study [7,8,9,18]. The target was the implementation of an experimental setup, aiming to prove whether a conservative PDES approach is feasible inside SIMSAT. The implementation of an optimistic approach was considered as not feasible due to the fact that emulator existence induces extremely high complexity and large memory requirements.

The first step regarding the implementation was the transformation of the scheduler to support multiple threads running concurrently having a separate input event list, without performing changes in the SMP2 adapter of the simulation objects, [18]. The next step was the introduction of an ad-hoc mechanism to support "local time" in each logical process using the global simulation time; thus being able to correlate and synchronize events running in different logical processes, [16].

The proof of concept was realized by the concurrent execution of a REFA based generic spacecraft, [3, 4], in a LP, and at the same time having specific parts and events of the system running in different LPs; thus requiring less time to execute than in the pure sequential setup. During the experimentation two major issues were highlighted.

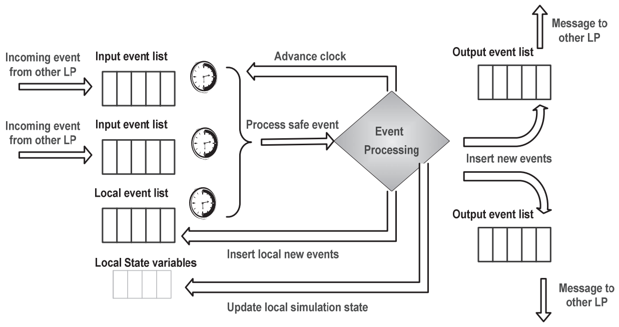

Figure 1. PDES Logical Process Structure.

The first one was lack of events in a REFA designed spacecraft. REFA assumes sequential execution of events, thus direct function calls between models dominate, in order to avoid event scheduling and associated data exchanging overheads. In order to take advantage of a PDES scheduler, a methodology for selecting REFA function calls and altering them to event based calls is vastly required. The second issue was related with the emulator running the On Board Software of the spacecraft. Most of the simulated models are tightly coupled with the Processor Module that executes the OBSW. This fact greatly reduces the chance to achieve a great degree of parallelization especially inside the Data Handling System.

Our approach proved that the PDES concept is feasible inside SIMULUS infrastructure [18]. Our next step was to think about a methodology and the required tools able to facilitate PDES application inside operational simulators, as well as explore further performance optimization threads

3. PERFORMANCE OPTIMIZATION FRAMEWORK (POF)

The POF is considered as an additional (and optional) stage before the final integration of the operational simulator assembly. The integrator follows a methodology having two principal performance optimization directions: Qualitative Approach and Quantitative Approach. Both directions require a Performance Analysis Entry Point in order to perform a startup analysis of the operational simulator that is under investigation and develop a set of benchmark tests. The latter tests define the "figure of merit" for comparisons between successive improvements and modifications aiming at performance optimization.

The framework starts with a start-up phase that involves installation of tools and utilities useful for generation of "mission" scenarios, configuration and acquisition of performance related information. During the present study the following tools have been designed and developed as a proof-of-concept, in order to support the POF:

- Performance Indicator Tools (PIT): PIT-IPM (measurements inside the spacecraft models), PIT-SPM (measurements inside simulator kernel), PIT-EPT (interfacing of external profiling tools [15]), PIT-ERS (generation of event and function call graphs [14])
- Configuration Application; i.e. user interface supporting the configuration of the PIT [13]
- Open Office Macro Library Functions and Templates [13].

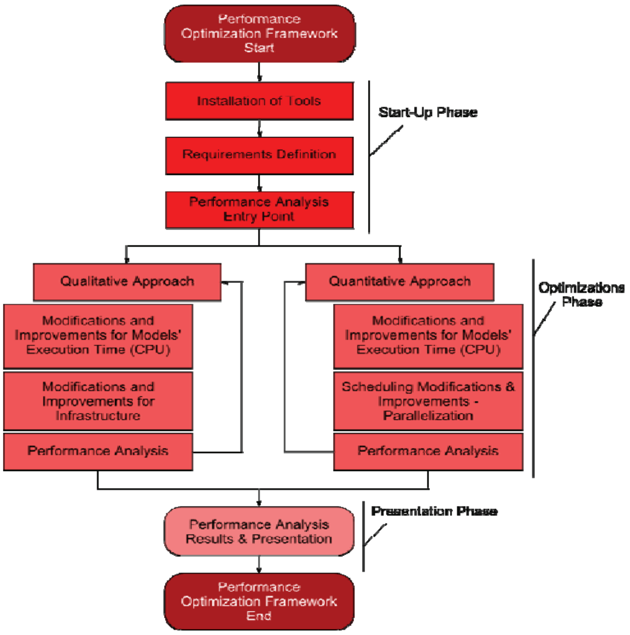
Performance Optimization Framework Start
Installation of Tools
Requirements Definition
Performance Analysis Entry Point
Start-Up Phase
Qualitative Approach
Quantitative Approach
Optimizations Phase
Modifications and Improvements for Models' Execution Time (CPU)
Modifications and Improvements for Infrastructure
Performance Analysis
Modifications and Improvements for Models' Execution Time (CPU)
Scheduling Modifications & Improvements - Parallelization
Performance Analysis
Performance Analysis Results & Presentation
Presentation Phase
Performance Optimization Framework End

Figure 2. POF Workflow.

The Requirements definition step is expected to be augmented by assumptions and suggestions as a result of the Performance Analysis Entry Point. The latter is an initial profiling of the Operational Simulator without having specific optimization targets, aiming at acquiring a general sense of the overall performance. Moreover, during this step the investigator must design and develop a set of benchmarking tests useful to define a measure of achieved optimizations.

The main part of the POF is the Optimization Phase separated into two independent strategies: a Qualitative and a Quantitative Approach. Obviously, both strategies aim at the same target which is the performance optimization, following completely different tools, procedures and concepts. The major diversification factor between the two approaches is the acquired information by means of how this can be utilized to drive to specific optimization fixes. The Qualitative Approach is based on the PIT-EPT tool, performing measurements using the o-profile application [15]; i.e. CPU profiling measurements based on statistical methodology. In contrast, the Quantitative Approach is based on the PIT-ERS tool, acquiring exact timing measurements and generate event and function call graphs based on the Graphiviz application [14]. In any case, both approaches are valuable as they provide complementary information. Finally, the Presentation Phase is dedicated to the generation of a final presentation including the results of the benchmark tests' execution (as defined in the Performance Analysis Entry Point) in order to demonstrate the achieved performance optimization.

4. CONCLUSION

The present paper briefly presented concepts, experimentation and results regarding performance optimization of SIMULUS based operational simulators. Our first target was to elaborate on a proof-of-concept for a PDES simulation engine inside the simulator's kernel. The second target involves the idea of a Performance Optimization Framework (POF) and the associated prerequisites, as an optional stage before the final assembly of an operational simulator. Our next step shall be the evaluation of the concepts in a real-mission operational simulator (i.e. GAIA).

5. REFERENCES

[1] V. Reggestad, D. Guerrucci, P.P. Emanuelli, D. Verrier (2004). Simulator Development: the flexible approach applied to Operational Spacecraft Simulators, SpaceOps, May 2004.

[2] Nuno Sebastião (OPS-GIC), Vemund Reggestad (OPS-GDA), David Verrier (OPS-GDS), "ESOC Simulators: Past, Present and Future", OPS-G Forum, 15 Jan 2008

[3] ESA/ESOC EGOS-SIM-REFA-SUM-1002, "Operational Simulator Development Guide",

[4] ESA/ESOC EGOS-SIM-REFA-SUM-1003, "General Simulator Developers Guidelines"

[5] ESA/ESTEC, ECSS-ETM-40-07, "Simulation Modeling Platform", 2008-07

[6] Ian McGregor, "The Relationship Between Simulation and Emulation", Proceedings of the 2002 Winter Simulation Conference

[7] ESA/ESOC DOPS-STU-SOW-1002-OPS-GDA, "Linux and Multi-Core Processor Technology for Simulators – Statement of Work", 2009

[8] ESA/ESOC DOPS-STU-TN-1002-OPS-GD, "TN1: Simulators Designer Guide for Performance Issues", 2011-09

[9] ESA/ESOC DOPS-STU-TN-1003-OPS-GD, "TN2: The Multi Core and Performance Project Report", 2011-09

[10] ESA/ESOC DTOS-SST-TN-0698-TOS-GIC, "Application of a Prototype Concurrent Scheduler Technical Note", 2003-06

[11] T. J. Schriber and D. T. Brunner, Inside Discrete Event Simulation Software: How it works and why it matters. In Proceedings of the Winter Simulation Conference, 2008.

[12] R. M. Fujimoto, "Parallel Discrete Ebent Simulation". In Communications of the ACM, Vol. 33 No. 10, Pages 30-53.

[13] ESA/ESOC DOPS-STU-SUM-101-OPS-GD, "PerfSim Software User Manual", 2011-01

[14] Graphviz, http://www.graphviz.org/

[15] Oprofile, http://oprofile.sourceforge.net/about/

[16] Jefferson, D.R., "Virtual Time", ACM Trans. Prog. Lang and Syst. 7, 3 (july 1985), 404-425

[17] J.S.Steinmain, "The WarpIV simulation Kernel", Proceeding of the Advanced and Distributed Simualation Workshop, 2005, pp.1-9

[18] Vemund Reggestad, Nikolaos-A.I.Livanos, Pantelis Antoniou, Ioannis E. Venetis, "Operational Simulators Going Parallel: From a Dream to a Concrete Concept", In Proc. of the 2011 DASIA, San Anton, Malta, May 2011.