Large-scale EXecution for Industry & Society

Deliverable D5.3

HW/SW integration optimization supporting Aeronautical use cases

Co-funded by the Horizon 2020 Framework Programme of the European Union
Grant Agreement Number 825532
ICT-11-2018-2019 (IA - Innovation Action)

DELIVERABLE ID | TITLE	D5.3 | HW/SW integration optimization supporting Aeronautical use cases
RESPONSIBLE AUTHOR	Donato Magarielli (Avio Aero)
WORKPACKAGE ID | TITLE	WP5 | Aeronautics Large-scale Pilot
WORKPACKAGE LEADER	Avio Aero
DATE OF DELIVERY (CONTRACTUAL)	30/06/2021 (M30)
DATE OF DELIVERY (SUBMITTED)	08/07/2021 (M31)
VERSION | STATUS	V1.0 | Final
TYPE OF DELIVERABLE	R (Report)
DISSEMINATION LEVEL	PU (Public)
AUTHORS (PARTNER)	Atos; IT4I; LRZ; LINKS
INTERNAL REVIEW	Vytautas Jancauskas (LRZ); Thierry Goubier (CEA)

Project Coordinator: Dr. Jan Martinovič – IT4Innovations, VSB – Technical University of Ostrava
E-mail: jan.martinovic@vsb.cz, **Phone:** +420 597 329 598, **Web:** https://lexis-project.eu

DOCUMENT VERSION

VERSION	MODIFICATION(S)	DATE	AUTHOR(S)
0.1	Definition of the document structure	05/05/2021	Donato Magarielli, Ennio Spano, Mirko Cortillaro (Avio Aero); Solofo Ramangalahy, Laurent Ganne (Atos); Jan Martinovic, Tomas Martinovic, Tomas Karasek, Jan Křenek (IT4I); Rubén Jesús García-Hernández (LRZ); Paolo Savio (LINKS); Michele Marconcini, Francesco Poli (UniFi)
0.2	Writing of connecting narrative and integration of contribution from UniFi	07/06/2021	Donato Magarielli (Avio Aero)
0.3	Integration of Ennio's contribution	09/06/2021	Donato Magarielli (Avio Aero)
0.4	General review	15/06/2021	Paolo Savio (LINKS); Donato Magarielli (Avio Aero)
0.5	Final review based on comments from internal reviewers	28/06/2021	Donato Magarielli (Avio Aero); Vytautas Jancauskas (LRZ); Thierry Goubier (CEA)
1.0	Final check	02/07/2021	Katerina Slaninova (IT4I)

GLOSSARY

ACRONYM	DESCRIPTION
API	Application programming interface
BD	Big data
CAE	Computer-aided Engineering
CFD	Computational fluid dynamics
CPU	Central processing unit
DAM	Dynamic Allocator Module
DDI	Distributed Data Infrastructure
DT	Digital Technology
FPGA	Field Programmable Gate Arrays
GPU	Graphics processing unit
HPC	High performance computing
HW	Hardware
I/O	Input/Output
ISV	Independent Software Vendors

KPI	Key Performance Indicator
MPI	Message Passing Interface
NVME	Non-Volatile Memory Express
OPENMP	Open Multiprocessing
PGI	Portland Group Inc
RAM	Random Access Memory
SPH	Smoothed-particle hydrodynamics
SW	Software
VOF	Volume of fluid
WP	Work Package

TABLE OF PARTNERS

ACRONYM	PARTNER
Avio Aero	GE AVIO SRL
Atos	BULL SAS
AWI	ALFRED WEGENER INSTITUT HELMHOLTZ ZENTRUM FUR POLAR UND MEERESFORSCHUNG
BLABS	BAYNCORE LABS LIMITED
CEA	COMMISSARIAT A L ENERGIE ATOMIQUE ET AUX ENERGIES ALTERNATIVES
CIMA	CENTRO INTERNAZIONALE IN MONITORAGGIO AMBIENTALE - FONDAZIONE CIMA
CYC	CYCLOPS LABS GMBH
ECMWF	EUROPEAN CENTRE FOR MEDIUM-RANGE WEATHER FORECASTS
EURAXENT	MARC DERQUENNES
GFZ	HELMHOLTZ ZENTRUM POTSDAM DEUTSCHESGEOFORSCHUNGSZENTRUM GFZ
ICHEC	NATIONAL UNIVERSITY OF IRELAND GALWAY / Irish Centre for High-End Computing
IT4I	VYSOKA SKOLA BANSKA - TECHNICKA UNIVERZITA OSTRAVA / IT4Innovations National Supercomputing Centre
ITHACA	ASSOCIAZIONE ITHACA
LINKS	FONDAZIONE LINKS / ISTITUTO SUPERIORE MARIO BOELLA ISMB
LRZ	BAYERISCHE AKADEMIE DER WISSENSCHAFTEN / Leibniz Rechenzentrum der BAdW
NUM	NUMTECH
O24	OUTPOST 24 FRANCE
TESEO	TESEO SPA TECNOLOGIE E SISTEMI ELETTRONICI ED OTTICI

TABLE OF CONTENTS

EXECUTIVE SUMMARY **6**

1 INTRODUCTION **8**

2 HW/SW OPTIMIZATION IN THE TURBOMACHINERY USE CASE **9**

2.1 COMPUTATIONAL PHASE 9

2.1.1 Porting of TRAF code on GPU-accelerated platform *9*

2.1.2 Checkpointing implementation *11*

2.1.3 Porting of TRAF code to FPGA-accelerated platform *13*

2.2 SIMULATION DATA VISUALIZATION 14

2.2.1 On-line monitoring *14*

2.2.2 Post-processing phase *15*

2.3 I/O OPERATIONS 16

2.3.1 Buffered-I/O option *16*

2.3.2 Profiling of the W/R operations *16*

3 HW/SW OPTIMIZATION IN THE ROTATING PARTS USE CASE **17**

3.1 PRE-PROCESSING PHASE 17

3.1.1 NVIDIA driver update *17*

3.1.2 SimLab software launch booster *18*

3.2 COMPUTATIONAL PHASE 18

3.2.1 Optimization of nanoFluidX execution *18*

3.2.2 Deployment of nanoFluidX on multiple GPU-accelerated nodes *22*

4 SUMMARY **23**

REFERENCES **24**

LIST OF TABLES

TABLE 1 BENCHMARK RESULTS AFTER THE FIRST FEW OPTIMIZATIONS 10
TABLE 2 BENCHMARK RESULTS AT THE END OF THE OPTIMIZATION ACTIVITY 11
TABLE 3 "REDUCED" TEST CASE PERFORMANCE FIGURES 11
TABLE 4 TEST CASES SIMULATING SINGLE WHEEL OIL IMPACT 19
TABLE 5 TOTAL NUMBER OF PARTICLES FOR EACH MODEL IN THE SINGLE WHEEL TEST CASE 19
TABLE 6 NUMBER AND DIAMETER USED FOR PRELIMINARILY MODELLING THE PLANETARY GEARBOX 20

LIST OF FIGURES

FIGURE 1 LONG-RUNNING JOB EXECUTION 12
FIGURE 2 TRAF HEAPPE JOB EXECUTION 13
FIGURE 3 POST-PROCESSING OF ALL REQUIRED VARIABLES PRESSURE (TOP LEFT) MOMENTUM (TOP RIGHT) DENSITY (BOTTOM LEFT) STAGNATION ENERGY (BOTTOM RIGHT) 15
FIGURE 4 PRESSURE IN CYLINDRICAL SLICES WITH DIFFERENT RADIUS VALUE 15
FIGURE 5 SINGLE WHEEL TEST CASE 18
FIGURE 7 ROTATING PARTS TEST CASES EVOLUTION IN LEXIS PROJECT 20
FIGURE 7 ENOVAL MODEL GEOMETRY 20
FIGURE 8 AXIALLY REDUCED GEOMETRY FOR OIL INVESTIGATION 21
FIGURE 9 PRE-PROCESSING INVESTIGATION CARRIED OUT TO PREPARE ENOVAL MODEL 22

EXECUTIVE SUMMARY

The LEXIS project relies on three large scale pilot use cases to validate and deploy its technology and infrastructure improvements, assigning to each pilot its own work package. The Work Package 5 (WP5) that this deliverable refers to is dedicated to the Aeronautics Large-scale Pilot and relies on the advanced HPC/Cloud/BD platforms and techniques designed in WP2 (LEXIS Requirements Definition and Architecture Design) along with WP3 (Data system and data management) and WP4 (Orchestration and cloud services). Industrial usage of the LEXIS platform, for which WP5 is prototypical, triggered important design decisions, particularly within the security sector. Besides an elaborate security concept, the design of subsystems was adapted as well. For example, on demand the LEXIS Data system now generally encrypts data related to a certain Pilot or Use Case.

Focusing here on the industrial use cases implemented in the WP5, the aim of the Aeronautics Large-scale Pilot led by Avio Aero in LEXIS is to significantly improve the feasibility and exploitation of advanced Computer-Aided Engineering (CAE) numerical modelling capabilities able to predict the fluid-dynamic behaviour of aircraft engine critical components. From both a digital technology and business perspective, a marked step change is thus envisaged: faster and more accurate CAE analyses that exploit newly deployed HW/SW resources in an innovative cross-converged HPC/Cloud/Big Data environment enabling the implementation of greatly improved or newly designed CFD-based engineering methodologies. To meet this ambitious objective in the WP5, the industrial applicability of the LEXIS advanced engineering platform is under investigation through two aeronautical engineering case studies, one regarding turbomachinery and the other one referring to rotating parts representing gearboxes, both aimed to examine complex fluid dynamic behaviour in aeronautical engine critical components. Looking to the wider scenario of European jet engine manufacturers, the outcomes from such investigations and the innovations pursued in WP5 may allow to facilitate a significant increase of productivity and competitiveness.

In this context, the present document D5.3, that is the third technical deliverable of the whole WP5, is intended to illustrate the optimization tasks that have been performed during the deployment of the HW and SW resources here adopted in LEXIS to implement from a digital technology standpoint the two above-mentioned aeronautical engineering case studies. In the first case, the Turbomachinery case study, the main outcomes here presented in terms of optimization of the coupled HW/SW technologies used reveal significant improvements in the computational phase, in the simulation data visualization, and in the I/O operations of the underlying application workflow. In the second case, the Rotating parts use case, the activities here described have involved the pre-processing phase and the computational phase of the related application workflow and show promising results in the strategy of optimising the execution of the involved computational tasks.

Position of the deliverable in the whole project context

This deliverable is a product of WP5 activities carried out so far in *Task 5.1 - HW/SW Integration Requirements, Task 5.2 - Turbomachinery Use Case Set-up and Run* and *Task 5.3 – Rotating Parts Use Case Set-up and Run* and is framed in the context of the WP5 project tasks especially aimed to optimize the integration among HW and SW LEXIS technologies adopted in the LEXIS Aeronautics pilot to implement from a digital technology perspective the Turbomachinery and Rotating parts use cases.

As in any standard CAE analysis, the CFD simulations that the described Aeronautics use cases relies on include the following three stages:

- pre-processing,
- the execution of the analysis solver,
- post-processing of the results.

In the first case, the Turbomachinery use case, the pre-processing stage is out of the scope of the LEXIS project and is performed locally on the workstation of the end users, while the computational phase, strongly HPC-demanding and time-consuming, and the post-processing phase, requiring large-memory GPU-equipped visualization nodes, both need to leverage the advanced capabilities provided by LEXIS in terms of state-of-the-art supercomputing

resources, SW enhancements and HW/SW systems integration. On the other hand, in the framework of the Rotating parts case study, the CFD simulations that the Aeronautics Rotating parts case study relies on need to leverage the advanced capabilities provided by LEXIS in terms of state-of-the-art HW resources available and HW/SW systems integration skills throughout the three phases of CAE analysis above-mentioned.

The complexity and difficulty in the development of large-scale test beds, such as the two ones included in the LEXIS Aeronautics Pilot, arises out of obligation to combine the specification of the used application with the proper HW technologies and configurations aiming at the optimal mapping on the HW/SW architecture of the computing system used to execute it.

In this context, the present document will illustrate the main activities carry out so far to optimize the HW and SW technologies supporting a proper and efficient execution of the computational jobs involved in the considered Aeronautics use cases.

The key LEXIS partners contributing to this deliverable are:

- Atos as the responsible for the codesign tasks and work package leader (WP2),
- IT4I as one federated HPC service provider,
- LRZ as the other federated HPC service provider,
- LINKS as the coordinator of codesign tasks and work package leader (WP4),
- Avio Aero for the Aeronautics Pilot (WP5).

From a contractual standpoint, this report document has to be delivered at the end of M30.

As lined out above, the developments in WP5 up to the point of this deliverable, and the related platform usage by WP5 have strongly contributed to driving the platform co-design process, in particular with regards to security by design and secure data handling. As prime examples, the resulting general security and authentication/authorization strategies have been documented in Deliverables 4.5 [1] and 4.7 [2], and measures for secure data handling in the LEXIS Distributed Data Infrastructure will be presented in Deliverable 3.5 [3].

Description of the deliverable

The main purpose of this document is to provide an overview of the optimization tasks implemented from a mixed HW/SW integration perspective during the project activities so far conducted in WP5 for the digital technology (DT) deployment of the two use cases included in the LEXIS Aeronautics pilot.

The description of the optimization tasks here illustrated starts from the Turbomachinery use case and refer to the computational phase, simulation data visualization, and I/O operations of the underlying application workflow. Then follows an outline of the optimization activities carried out for the pre-processing phase and the computational phase of the Rotating parts use case.

Finally, we summarize the main outcomes from the above-mentioned optimization of HW and SW resources used in both the considered Aeronautics use cases.

1 INTRODUCTION

Reducing the fuel consumption of aircraft engines is a key requirement for players in the aeronautic industry. Significant efforts are underway to produce reliable turbo engine performance predictions, using physics-based multi-physics simulations to help anticipate problems typically encountered in the detailed design phases. This demands the adoption of CPU-intensive and time-consuming CAE simulations based on sophisticated numerical solvers.

From a digital technology and business perspective, Avio Aero goals are to achieve a marked step change: less time-consuming computational analyses that exploit newly designed, improved and/or tightly coupled, HW/SW components able to open the doors to the "real time" design approach for the engineering of turbines. Furthermore, the big data produced as a result will require proper solutions for quick data access, management, and post-processing.

Framed in this context, the exploitation in LEXIS of both next-generation HPC/Cloud/Big Data technologies and advanced CFD software solutions is enabling innovative and faster investigation strategies for the design and optimization of critical aircraft engine's components, such as low-pressure turbines and gearboxes, that in LEXIS are carried out through the Aeronautics Large-scale pilot. More specifically, the HW and SW resources required to run the CAE simulations involved in the Turbomachinery and Rotating parts use cases included in the Aeronautics pilot need to be very closely integrated and considerably optimized to allow properly and efficiently execute the related computational jobs.

In this deliverable the description of the tasks implemented (at M30) in LEXIS WP5 - Aeronautics Large-scale Pilot in terms of optimization of the coupled HW/SW technologies used in LEXIS is provided. In detail, this deliverable includes 3 main sections: 2 - HW/SW optimization in the Turbomachinery use case, 3 - HW/SW optimization in the Rotating parts use case, 4 - Summary.

Section 2 presents the activities focused to optimize the HW and SW technologies used in the Aeronautics Turbomachinery use case.

Section 3 describes the activities aimed to optimize the HW and SW technologies used in the Aeronautics Rotating parts use case.

Finally, Section 4 will highlight the final remarks of this deliverable, including the proposals for the future planned activities.

2 HW/SW OPTIMIZATION IN THE TURBOMACHINERY USE CASE

In this section, the optimization of the HW and SW resources adopted to support the digital technology implementation of the Aeronautics Turbomachinery use case is described.

2.1 COMPUTATIONAL PHASE

The computational phase of the Aeronautics Turbomachinery use case is based on TRAF code, a CFD application solver developed by the University of Florence to investigate fluid dynamics phenomena with a special focus on turbomachinery simulations. Extensively validated against several turbomachinery configurations, a high level of parallelization is provided in the solely CPU-based version of the code through a hybrid parallel programming model (MPI/multi-platform shared-memory parallel), but the detailed and increasingly complex CAE analyses performed during the Turbomachinery design process require a step change in terms of computational job duration. For this reason, development and optimization tasks for porting this code from a pure CPU-based computing platform to a GPU-accelerated one have been foreseen in LEXIS project, aimed to drastically reduce the execution time.

2.1.1 Porting of TRAF code on GPU-accelerated platform

At the beginning of the project, a number of tests were performed in order to assess the performance of the TRAF solver, with the main focus on the scalability of the parallel version (MPI + OpenMP) of the code on CPU-based platforms. These tests were carried out on the turbomachinery pilot use case provided by Avio Aero, which is representative of an unsteady Reynolds averaged Navier-Stokes simulation of a multistage aeronautical turbine, including two stages and the rear frame struts. The computational domain has been spatially discretized by means of a grid divided into different blocks. The domain consists of more than 700 mesh blocks for a total of about 1500 million cells, requiring an overall memory amount of about 1 TB.

After assessing the performance of the TRAF solver on CPU-based architectures, the focus shifted to evaluating the benefit that can be obtained by running TRAF on GPU-accelerated HPC platforms. The goal was to reduce the computational time by a factor of at least 5, by exploiting GPU acceleration. To this aim, a CPU-based calculation using one MPI process per mesh block was taken as the reference performance metrics. This target speedup is one of the KPIs set in the LEXIS project scope.

The University of Florence (UniFi) activity devoted to porting the TRAF code to GPU-accelerated platforms started before the LEXIS project kick-off: the CUDA Fortran programming language extension and the PGI/NVIDIA compiler are used to port the solver to NVIDIA accelerators.

The first step of the porting effort consisted of:

- data structure modification to define “device” (i.e.: GPU on-board memory) counterparts of all the main arrays used in the computation;
- allocation and initialization of these device arrays (done, as much as possible, at the beginning of the execution);
- modification of all subroutines and modules so that computational loops are offloaded into CUDA “cuf kernels” to be launched on the GPU;
- addition of conditional compilation directives, wherever needed, in order to preserve as much of the original structure of the source code as possible (thus allowing compilation for both CPU-based and GPU-accelerated architectures from the same source code base: a choice that greatly simplifies future code maintenance and development);
- continuous testing of the code on small tests, with a special focus on execution correctness, rather than on performance.

Once these actions had been carried out, a first preliminary release of the ported solver was completed.

The second step was to enable MPI (distributed memory) parallelism for the ported code in order to make it able to run on multiple GPUs (shared on the same host and/or distributed on distinct hosts): MPI parallel execution was already implemented in the TRAF solver, but some effort was needed to make it work correctly in the GPU-accelerated version of the code. The approach that was initially followed consisted in implementing the exchange of boundary data by:

1. copying array slices from device (GPU) to host (main) memory,
2. sending and receiving the slices between the interested MPI processes,
3. copying the received data back from host to device memory.

As expected, this host memory staging process had significant limiting effects on performance, but any improvement on this front was intentionally postponed, since implementation simplicity and execution correctness took priority.

At this point, with a functional release of the solver available, a code profiling and optimization activity could begin, first focusing on single GPU execution, then on MPI communications. In order to profile the code execution, the NVIDIA nvprof tool was used: before and after each code modification, the most time-consuming kernels could be identified and the benefit of each optimization could be measured.

After starting the optimization activity, it became apparent that the small tests were still needed to check the execution correctness, but were not enough to obtain a clear picture of the performance. A more representative benchmark was therefore defined, having mesh block dimensions comparable to those found in the turbomachinery pilot test case, but only featuring 18 blocks. This benchmark requires an overall memory amount of about 32 GB, thus filling the on-board memory of 2 GPUs of the IT4I Barbora HPC cluster. This benchmark was first performed on the Barbora cluster, after the first few code optimizations were completed. The results obtained with that TRAF version are shown in Table 1.

NUMBER OF NODES	MPI PROCESSES PER NODE	HARDWARE RESOURCES	EXECUTION TIME [s]	SPEEDUP
1	13	13 CPU cores	2975	(baseline)
2	1	2 GPUs	1030	2.89
2	3	2 GPUs	918.2	3.24
2	7	2 GPUs	759.8	3.92

Table 1 Benchmark results after the first few optimizations

The main KPI is the speedup, here defined as the CPU (baseline) to GPU computational time ratio. The number of MPI processes for the CPU-based calculation was chosen so that each process deals with a number of cells comparable to that of the biggest mesh block; 1 CPU core was assigned to each MPI process. On the other hand, the number of GPUs used in the accelerated calculations was selected to be compatible with the overall memory requirement.

The same benchmark was repeatedly used during the rest of the optimization activity, in order to check the progress status and assess the performance improvements. Among the several code changes, the use of multiple asynchronous execution queues was introduced in order to increase the operation concurrency. Many CUDA cuf kernels were replaced by equivalent OpenACC “parallel loops”, also taking advantage of some more flexible OpenACC directives. All the MPI communication code was overhauled: the manual host memory staging process was abandoned and CUDA-aware MPI was used.

At the end of the optimization activity, the benchmark was repeated one last time. The final results are shown in Table 2, where it can be seen that the maximum speedup has increased by about 154 % (from 3.92 to 9.96).

NUMBER OF NODES	MPI PROCESSES PER NODE	HARDWARE RESOURCES	EXECUTION TIME [s]	SPEEDUP
1	13	13 CPU cores	2975	(baseline)
2	1	2 GPUs	298.6	9.96
2	3	2 GPUs	353.0	8.43
2	7	2 GPUs	337.7	8.81

Table 2 Benchmark results at the end of the optimization activity

The above-described benchmark was useful to assess the performance improvements obtained with the applied code optimizations. After this assessment, an even more realistic test was needed, based on the turbomachinery pilot use case. However, the total combined on-board memory capacity of all the GPUs available on the Barbora cluster was not enough to accommodate the “complete” case. It was therefore decided to define a new case, which corresponds to a quarter of the original domain: this “reduced” case approximately fills the on-board memory of 16 Barbora cluster GPUs.

In the “reduced” test case, the baseline calculation was performed by 108 MPI processes on 108 CPU cores (on 3 nodes), with no GPU acceleration. A second setup doubled the number of used CPU cores (216 on 6 nodes), by activating 2 OpenMP threads per MPI process, while still using the same number of MPI processes (108), again without GPU acceleration. On the other hand, the GPU-accelerated calculation employed 16 GPUs (and 16 supporting CPU cores) on 4 nodes, by running 16 MPI processes (one MPI process on each GPU). The tests were carried out by performing the unsteady simulation for a few time steps and then measuring the average net computational time per step. The performance results for this test are shown in Table 3.

NUMBER OF NODES	MPI PROCESSES × OpenMP THREADS	HARDWARE RESOURCES	NET TIME PER STEP [s]	SPEEDUP
3	108	108 CPU cores	689.4	(baseline)
6	108 × 2	216 CPU cores	399.0	1.73
4	16	16 GPUs	132.0	5.22

Table 3 “Reduced” test case performance figures

It is apparent that the achieved speedup (5.22) fulfils the minimum goal (5) set for the porting activity, although it is smaller than the best speedup obtained in the benchmark. The different number of MPI processes between the pilot test case and the smaller-sized benchmark has probably to be taken into account: the greater amount of MPI communications involved in the pilot test case could explain the performance reduction. Further enhancements and code optimizations may even more improve these figures in the future.

2.1.2 Checkpointing implementation

Each HPC cluster has reserved computational nodes for specific queues with a different execution purpose (testing, production, long runs, etc.) with different wall-time limitation. Typically, max wall-time for execution at a long queue at the HPC cluster is in the order of days. This raises the question of what to do when the wall-time limit for

the long queue is not enough for calculation? The solution is to divide the long-running job into small jobs and setting up dependencies between them. This solution has one problem - the job must implement some kind of checkpointing mechanism to restore computation in another small job. Figure 1 shows the execution of a long-running job on an HPC cluster:

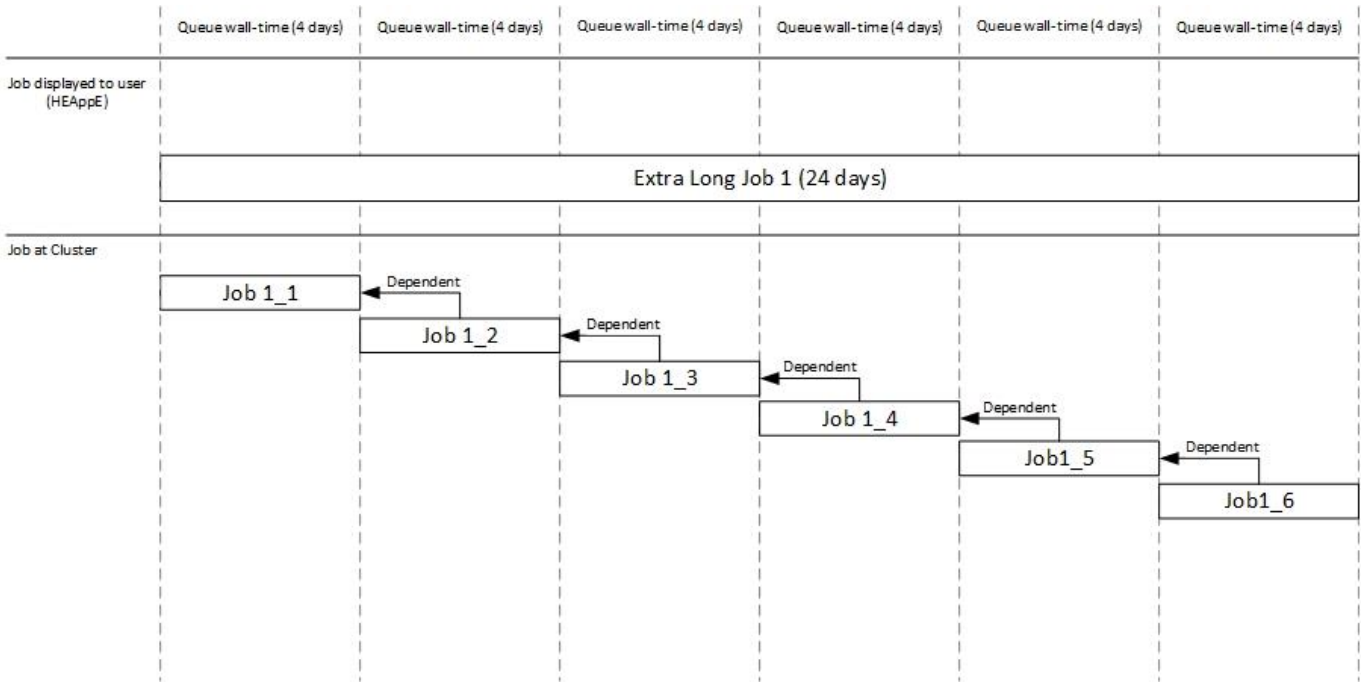

Figure 1 Long-running Job execution

Checkpointing in HPC cluster

Checkpointing functionality is important for recovery calculation from previously calculated status (not needed to calculate from the scratch again). Checkpoints are created for these two possible scenarios:

- recovery application from failure,
- exceeded wall-time limit of queue.

In our case, TRAF execution has implemented a periodic checkpointing mechanism (create checkpointing files every specific period) for the two possible scenarios mentioned above. Thanks to the implemented checkpointing mechanism, it is possible to execute TRAF as a sequence of smaller jobs, where each partial job is dependent on the previous one. A consequence is that, if one such job fails, all follow-up jobs will fail. Still, compared to a single job, all jobs before the failed one do not have to be repeated: the restart will occur at the first failed job of the sequence.

Checkpointing implementation in HEAppE Middleware

HEAppE Middleware functionality [4] was extended to support the execution of long-running jobs in HPC clusters. For the execution of the long-running job, it is necessary to specify HEAppE API parameters ("IsExtraLong" setting and an increase of the wall-time limit to the application execution wall-time). HEAppE does automatically splitting the long-running job into small dependent HPC scheduler jobs with a maximum queue wall-time limitation. For storage optimization, HEAppE creates symbolic links to data from the previously executed job. This feature has been encapsulated for end-user adoption purposes and provides them the ability to simplify HEAppE API usage.

Checkpointing implementation in the workflow

To manage checkpoints, the workflow relies on the HEAppE API to regularly monitor the list of files produced by TRAF during its execution. For each file, HEAppE provides the last modification date of the file.

A software component using this information was implemented to:

- identify checkpoints among these produced files, from their name (patterns *udft*_t*.dat* and *u_t*.dat*),
- when a checkpoint file hasn't been modified for more than 5 minutes, consider this file is complete, and store this checkpoint file in LEXIS Distributed Data Infrastructure (DDI) in a dataset grouping files by same pattern value **_t([0-9]+).dat*.

In case of failure in the cluster running TRAF, the workflow will switch to an "on failure" branch where it will call the Orchestration Service Dynamic Allocator Module (DAM) - WP4, to find an up and running HPC location where to create a new TRAF HEAppE job, to which the input dataset and checkpoints dataset from the previous run will be transferred.

Figure 2 shows a simplified version of the workflow subset implementing this:

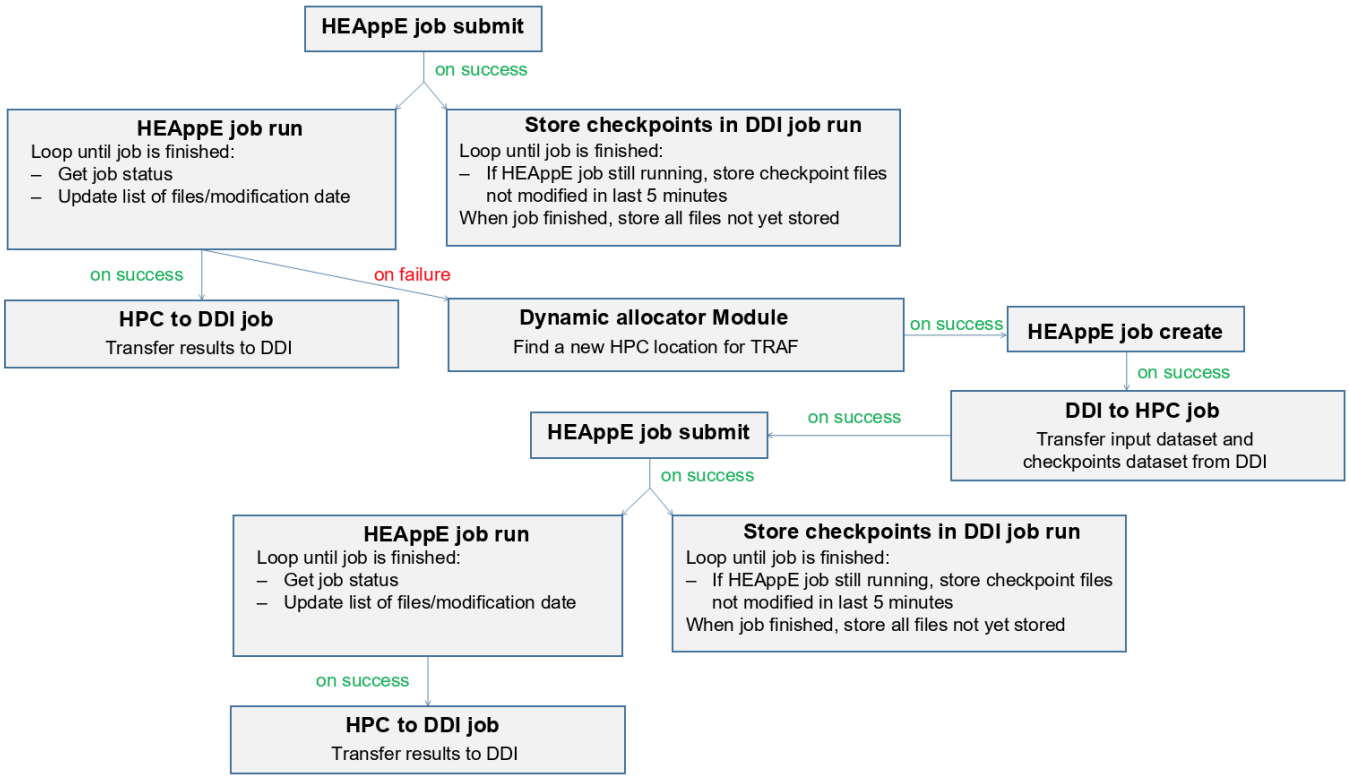

Figure 2 TRAF HEAppE job execution

2.1.3 Porting of TRAF code to FPGA-accelerated platform

In order to port the TRAF code from solely CPU-based platform to GPU-accelerated ones, some parts of TRAF code have been rewritten using the OpenACC and Cuda Fortran programming languages.

With the aim of enabling compute operations also on a FPGA-accelerated HPC platform, a new rewriting of the code is required and may be based on the OpenCL programming language.

OpenCL is very similar to CUDA software framework and allows the writing of C-like kernels that can be synthesized into an FPGA bitstream.

A quick analysis of the TRAF source code showed that the solver usually relies on single-precision operations; however, some portions of the code need to perform calculations in double-precision to obtain accurate results.

Compared with the GPUs, FPGAs are more flexible to be programmed from a computational perspective, because custom data paths can be easily defined; however specific software libraries (already available, but, generally

speaking, slower) are required to allow double-precision operations since the arithmetic logic units in FPGAs (so-called DSP slices) are limited to single precision floating point.

The residual smoothing subroutine is the most HW-intensive one in TRAF code and can be a good candidate to be ported on FPGA-based platforms. It is written in CUDA-aware Fortran, it represents 200-300 lines of code and aims at solving tridiagonal system of equations (Thomas' algorithm) that should require not more than 32 GB RAM. The internal memory of FPGAs is quite small (some MBs of dual port RAM and registers embedded in the programmable logic), so to successfully support the porting of the residual smoothing subroutine, access to external DDR memory to allow proper compute operations is required.

Thomas algorithm is essentially serial (the data set is processed twice, at first forward and then backwards to get the solutions, with data dependency limiting the possibility of an aggressive loop unrolling); however, parallel implementations have been proposed in literature and will be investigated.

A quick analysis of the FORTRAN routine showed that this problem is essentially memory bound: the routine sweeps the multi-dimensional arrays (four dimensions, so four levels of nested loops) reading and writing back the results after easy numerical operations, so the number of parallel workers is practically limited by how fast we can feed the computational pipeline.

The BITTWARE 520n FPGA boards that have been selected for this project have 32 GB of external SDRAM memory divided into 4 memory banks, so the memory requirements of the residual smoothing routine operations should be met.

However, to allow the computations on the FPGA, the data must be transported from the host memory to the external memory and then from the external memory to the internal FPGA's one and, at the end of the algorithm, all the way back to the host memory for the next iteration. If not efficient, this data transfer process may introduce a potential bottleneck in the computational performance, so to not lose the potential benefits of the FPGA acceleration, the communication between the two memories should be as fast as possible.

Currently, the smoothing routine is being translated from Fortran to OpenCL and in the near future we plan to synthesize the obtained OpenCL code. Moreover, a small test dataset has been built in order to assess the potential speed-up difference with respect to the CPU-only and the GPU-accelerated versions of this routine.

2.2 Simulation data visualization

The visualization of data from the CFD simulations of the Turbomachinery use case include the following two phases:

- the on-line monitoring, that consists in real-time displaying the convergence history (residuals) for each basic computational step,
- the post-processing phase, that process the results from the computational phase in order to display the engineering variables of interest.

The activities aimed to optimize the execution of the above phases are here described.

2.2.1 On-line monitoring

The LEXIS portal supports the visualization of image files contained in datasets stored in the Distributed Data Infrastructure (DDI); the images are refreshed automatically at regular intervals to support dynamically-updated datasets.

We leverage that functionality by using open source software tools like ImageMagick® [5] and gnuplot [6] to generate an image containing graphs of the current progress of the workflow at regular intervals and uploading these files to the DDI at a pre-specified path.

The user can visualize the graphs in the LEXIS portal while the workflow is running, receiving immediate feedback about the workflow convergence history.

2.2.2 Post-processing phase

The post-processing of simulation results is usually a manual task, performed when the user is visualizing the obtained results. In many cases, and the Aeronautical use case in LEXIS is one of them, automatic off-line visualization could bring added value to the whole process. This added value lies in:

- visualization of data during the computation,
- visualization of large datasets,
- comparison of results obtained from the same model but with different boundary conditions.

In LEXIS, for off-line or automatic post-processing of the results, the visualization tool ParaView® [7], running in batch mode after executing python scripts, was exploited. The adopted scripts were created based on the requirements of the Turbomachinery use case, for which all the engineering variables that should be visualized - Pressure, Density, Momentum and Stagnation Energy - (see Figure 3 for reference) were defined.

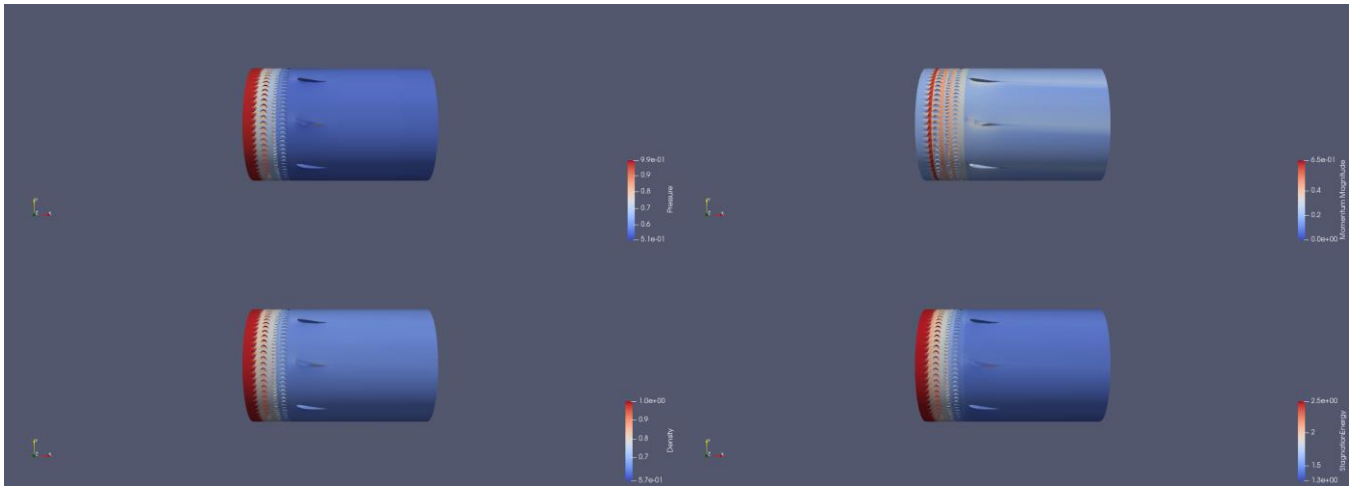

Figure 3 Post-processing of all required variables Pressure (top left) Momentum (top right) Density (bottom left) Stagnation Energy (bottom right)

Moreover, slices through computational domain were defined for better analysis of the obtained results. Since in Turbomachinery use case the computational domain has a cylindrical shape, the slices are cylindrical (see Figure 4), with different radiuses.

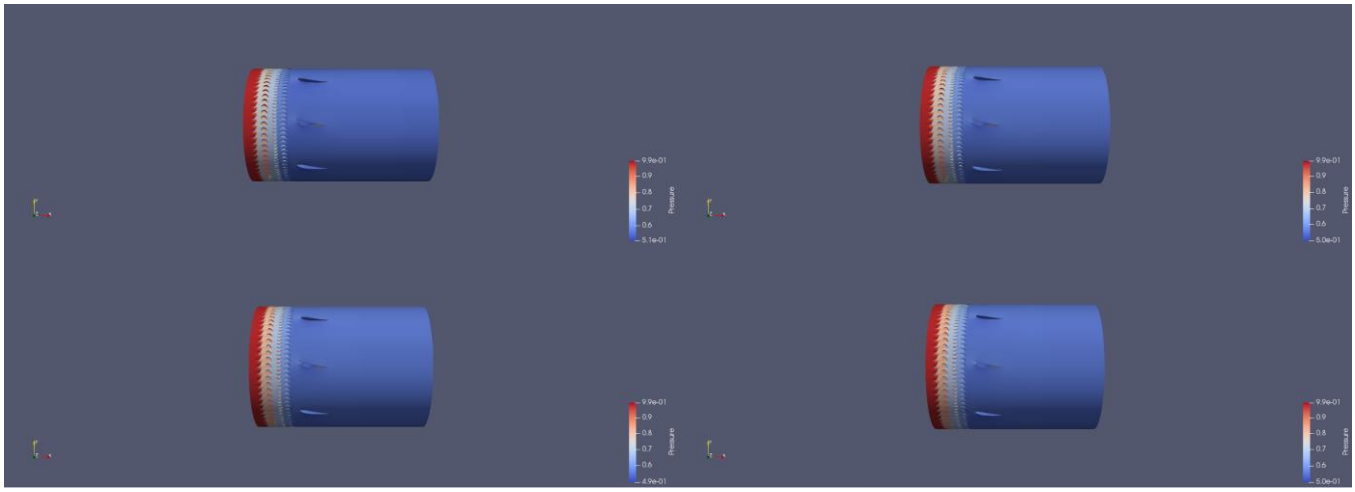

Figure 4 Pressure in cylindrical slices with different radius value

Python scripts were written in such a way that they could be easily modified to add additional features. All the user defined variables which are used to manage the post-processing are placed at the very beginning of the script. In the future, if necessary, this script could be extended by adding other quantities to visualize, showing slices through the domain or planar slices, instead of cylindrical ones. This could be done by simply adding blocks of Python code representing required functions of the Paraview® tool.
All the scripts were deployed and used on the visualization nodes of the Barbora supercomputer.

2.3 I/O operations

To further optimize the execution of TRAF computational jobs by getting maximum performance out of the used hardware, the write and read operations of TRAF code have been analysed in depth. The outcomes from such analysis are here illustrated.

2.3.1 Buffered-I/O option

The initial version of the CPU-enabled TRAF code showed low performance on LRZ systems. Under guidance from LRZ staff, the issue was found: the LRZ's Linux Cluster and SuperMUC-NG use an I/O buffer size of 8 MB and 16 MB respectively to optimize network usage (standard values for local disks in non-HPC systems are 512 B to 4 KB). However, by default the Intel® Fortran compiler uses non-buffered I/O (this ensures that changes of a file by a different processor are visible immediately by all processors and helps with coherency, but comes at an especially high cost in these HPC systems).

Since our code can safely use buffered I/O, we added "-assume buffered_io" and "FORT_BLOCKSIZE" to the runtime, providing a speedup of the initial loading of the input files from 30 minutes to a minute.

We also requested that buffered I/O was made the default for all LRZ supercomputing users. The LRZ documentation [8] was updated to include our findings.

2.3.2 Profiling of the W/R operations

The profiling of the CPU version of TRAF was done to estimate the read and write times, mostly at the beginning and end of a job. For a job which runs for approximately 24 hours, the system is using the first 40 minutes for reading the data and preparing for the computation, then computes for 21 hours with an average of 82% utilization of the CPU resources and it performs the post-processing of results for a little over two hours. The setup phase and post-processing phase are fixed in terms of computational time for three hours. That means if the IT4I resources were used along with the large maximum job walltime of 144h, it would take up 2% of the computational time. This was deemed sufficient at current phase by Avio Aero, so no further steps to optimize setup and post-processing phase were taken.

3 HW/SW OPTIMIZATION IN THE ROTATING PARTS USE CASE

This section describes the optimization of the HW and SW resources adapted to support the implementation of the Aeronautics Rotating Parts use case from a digital technology perspective.

3.1 Pre-processing phase

Compared to the standard Volume Of Fluid (VOF) approach, the pre-processing phase supporting the newly designed CFD-based application workflow included in the Aeronautics Rotating Parts use case aims to be faster because it doesn't need to generate a mesh in the classic sense from the user input project, but just imports the geometric shapes and fills them with particles of the required size and properties. However, the simple installation and the use of an advanced simulation algorithm are not enough to innovate the actual design flow, because to run at optimum performance, such an algorithm needs to be properly coupled with the state-of-the-art computing hardware available in LEXIS. In this regard, the activities aimed to optimize the digital technology implementation of the pre-processing phase are here described.

3.1.1 NVIDIA driver update

The pre-processing phase of the Rotating part use case is performed with the Altair SimLab™ [9] software which is part of the HyperWorks Suite [10].

As is often the case with ISVs (Independent Software Vendors), the software stack is complex: making use of a lot of libraries and supporting several platforms.

A specific issue with SimLab® is that remote visualization tools, VirtualGL in particular, are not tested nor officially supported.

Nevertheless, Altair provides a script (diag_tool) which produces a report with the list of hardware and software components to compare it with supported configurations:

- HW: processor, memory, network, graphic cards
- SW: distribution, kernel version, graphical libraries, X11 and OpenGL

The granularity of detail is high. For example, initially the NVIDIA driver which was installed on the nodes was version 450.51.06, whereas the version indicated in the "Hardware Recommendations and Certifications" manual is 450.57 (or higher). This needed a maintenance period for the Barbora HPC cluster at IT4I in order to upgrade the driver (to version 460.56).

The tests have been performed on the visualization nodes of the Barbora cluster at IT4I. These nodes have NVIDIA Quadro P6000 as GPUs, and uses the RHEL Linux distribution.

Compared to the HPC clusters, OpenStack clusters have the advantage of more flexibility for the installed software, since a user can install any software on the virtual machines. On the bare-metal nodes of Barbora, a certain amount of flexibility is provided by the use of user modules (via lmod), but this does not apply to system software like NVIDIA GPU drivers.

Using an OpenStack cluster would allow us to deploy the exact software stack recommended by Altair.

On the other hand, on OpenStack cluster, the deployment of the VM in the workflow can have significant delay for using the software, as Simlab™ distribution size is several gigabytes. Furthermore, use of virtual machines needs specific techniques like GPU passthrough or virtual GPU necessary to make use of the underlying GPU hardware. This adds further complexity to an already complex stack and moves us further from the configurations validated by Altair.

Thanks to this diagnostic tool and collaboration with Altair support, we have been able to use Simlab™ with remote visualizations tools VirtualGL [11] and TurboVNC [12].

3.1.2 SimLab software launch booster

SimLab software has been deployed on IT4I Barbora visualization nodes under the scratch file system where normally shared software packages are installed, to allow all end users with a specific project id to access and use them. During the launch of this software from that file system, users have experienced significant slowdowns due to the high number of I/O operations when this storage accessed by a lot of users. To solve this issue, two possible solutions have been identified with the support of IT4I technicians:

- RAMDISK -> Launch Simlab software using a local RAM disk of the Barbora node,
- NVMe Over Fabrics File System -> Attach a volume from an NVMe storage and mount it as a file-system on Barbora node and launch Simlab using this file system.

The second option would be recommendable because it would give a more significant improvement at the I/O level. The next step is to involve the technical support from the ALTAIR SW supplier to modify the SimLab application environment and to implement the launch script using the NVMe file system.

3.2 Computational phase

Designed to leverage the computing power of GPUs, the application solver used in the computational phase of the Rotating parts use case is able to achieve high levels of parallelism and excellent scalability on both single and multi-node GPU servers. The discretization process in the computational domain that is considered here, is based on a set of independent points, named particles. Different numbers of particles are used depending on the investigated air, liquid or mixed phase and test cases, requiring different running times. The more the simulation complexity increased according to the engineering needs for executing closer-to-reality simulations, the more the scalability on multiple GPUs over multiple HPC nodes is needed, allowing to identify the correct granularity of particles involved in the simulations so as to correctly and accurately predict the physical phenomena of interest. For running the application solver at its best over multiple GPUs, the fine-tuning of simulation jobs parameters has been implemented and is here briefly described.

3.2.1 Optimization of nanoFluidX execution

For capturing aeronautical gear-box flow physics elements, a novel software named nanoFluidX™ (by Altair Engineering) is being adopted within the scope of the LEXIS project, with the aim to obtain a significant step-change in prediction of key phenomena involved in the rolling gears, like windage effects and resistant torques both impacting on mechanical efficiency and power transmission levels.

In order to investigate the discretization level suited for well reproducing both the liquid and air phase motion, the code was tested first in some simplified cases, then moving on to more complex use cases. It should be noted that all the simulations will be checked and compared with experimental data. As a first test, an oil-jet directed towards a single wheel (refer to Figure 5) was reproduced. In fact, the aim of this test was to study the forces exchanged between the gear and liquid comparing results with torque experimental data as well as previous simulations performed with a numerical code based on Finite Volume Method (Ansys Fluent), which uses the Volume Of Fluid (VOF) approach to solve the multiphase flow.

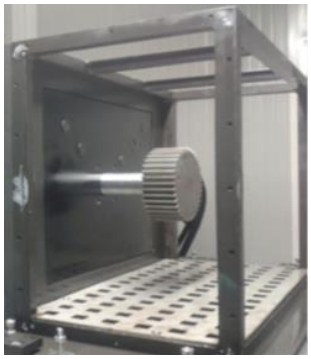

Figure 5 Single wheel test case

Simulations were run using IT4I hardware as reported in the Table 4: the key distinguishing element is clearly identified in the particle diameter that will populate both the fluid and solid domain and is at the base of the numerical simulation. Lower is

the diameter and higher is the solution accuracy as well as the overall model dimensions (evaluated in Millions of particles) and finally the computational time.

The highest model's dimensions recorded for this simple test case are referring to circa 150 M particles having run for 145 hours on DGX-2 machine at IT4I premises.

	FULL GEOMETRY			REDUCED GEOMETRY		
PARTICLE DIAMETER [mm]	**INITIAL PARTICLES NUMBER**	**GPUs HARDAWARE USED**	**SIMULATION TIME**	**INITIAL PARTICLES NUMBER**	**GPUs HARDAWARE USED**	**SIMULATION TIME**
0.24	3.8 M	DGX-1 V100		0.35 M	DGX-2	5' 45''
0.12	16 M	DGX-1 V100	3h 30'	1.4 M	DGX-2	17' 24''
0.06	65.8 M	DGX-2	28h 38'	5.7 M	DGX-2	2h 37'
0.04	149 M	DGX-2	~145 h			
0.03				23.2 M	DGX-2	36h 22'

Table 4 Test cases simulating single wheel oil impact

From the designer's point of view, a satisfactory condition can be identified in using models with 0.06 [mm] particle diameters that represent a good compromise between accuracy and acceptable computational running time. This value corresponds to circa 1/20 of oil jet diameter used in the test rig.

As a second test, the SPH model developed to simulate the air motion inside the single wheel test case (refer to Table 5) is discussed. A big volume of fluid is involved since the entire test chamber of the experimental rig has been numerically reproduced: the computational domain is always based on a single wheel, clamped on a shaft, enclosed within a test box. The particles total number for each model are reported in the Table 5.

MODEL	PARTICLE DIAMETER [mm]	PARTICLES TOTAL NUMBER
Coarse	2	12.9 M
Mid	1.5	30.5 M
Fine	1.2	59.3 M

Table 5 Total number of particles for each model in the single wheel test case

The total amount of particles is not extremely high but nevertheless the computational time of the fine grid solution needed 171 hours to simulate a flow time of 1s and running on 16 GPUs DGX-2 cluster.

In summary, the most critical test cases run up to now were referringto:

- Oil-only simulation, that generated a numerical model of circa 150 Million of particles.
- Air-only simulation, that needed a long computational time to guarantee the achievement of steady state conditions: 171 hours (about 1 week) inside the single wheel chamber.

The hardware in question was constituted of 16 V100 GPUs.

The development activities being carried on in the LEXIS project have foreseen to investigate, as incremental step, the flow physics involved in a double wheel environment (refer to). This test case represents a step beyond for approaching closer-to-reality conditions. The methodology defined to approach this environment has been based on the adoption of so called “reduced” numerical models that proved to be sufficiently accurate in reproducing key phenomena linked to the meshing of the two wheels. For reduced models, we mean a model in which the complete wheel is not reproduced but only a sector of it (typically covering less than 90 degrees).

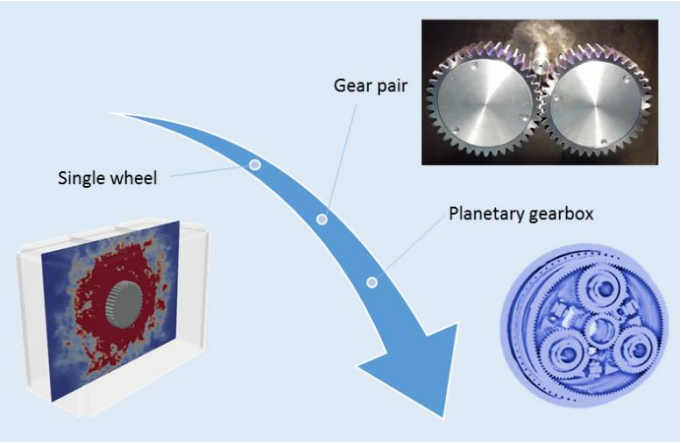

Figure 6 Rotating parts test cases evolution in LEXIS project

This has been a very important test case from a physical perspective since allowing to understand how nanoFluidX® code is performing while two wheels are meshing each other, but not really demanding from a hardware point of view.

As the conclusive step, the planetary gearbox has to be modelled, a geometry very similar to industrial applications, which was measured at the THT Lab in the framework of ENOVAL European project.

This test case is really the biggest challenge in terms of air volumes involved and mechanical complexity. The SPH geometrical model is reported hereafter in Figure 7:

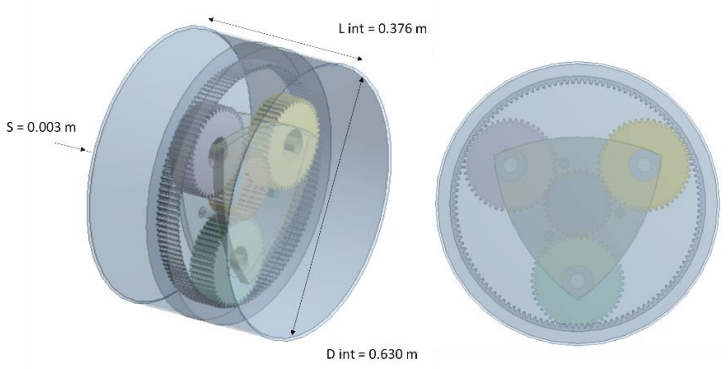

Figure 7 Enoval model geometry

Three points (pipes) of oil jets have been inserted into the gearbox in order to promote the lubrication of the wheels (solar and satellites) meshing each other.

First attempts at modelling are summarized in the Table 6. To guarantee sufficient accuracy of the numerical air+oil mixture simulation, a particle diameter of 0.07 [mm] should be used. Unfortunately, this choice implies involving billions of particles and this is not currently computationally feasible.

PARTICLE DIAMETER [mm]	PARTICLES REQUIRED [Million]
1.4 (1 particle per jet diameter)	4.12
0.7 (2 particles per jet diameter)	33
0.35 (4 particles per jet diameter)	264
0.07 (20 particles per jet diameter)	Not Applicable

Table 6 Number and diameter used for preliminarily modelling the planetary gearbox

Hence in terms of simulation, the overall strategy has to be changed. The best approach should have been identified:

1. Reproduce the real air volume that is significant but possible to be reproduced (this means to consider the right geometrical box dimensions).
2. Define an "axially reduced" geometry to investigate only the oil behaviour inside the gearbox, eliminating the axial parts not directly involved by the oil motion. Reduction in the domain dimensions will allow to increase the spatial resolution and accuracy of the newly developed SPH numerical model. Figure 8 shows how the reduced model will appear.

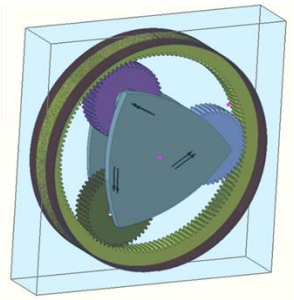

Figure 8 Axially reduced geometry for oil investigation

Before proceeding to set up the model for oil investigation, a detailed evaluation of how the numerical model dimensions change and increase as a function of the ratio of particles diameter to oil jet diameter has been carried out. The oil jet diameter corresponds to the ones put inside the gearbox and promoting the lubrication of the wheels (solar and satellites) meshing with each other.

The graphs in Figure 9 refer to the particles´ diameter to oil jet diameter ratio as the X axis. The higher is this value the better will be the simulation accuracy. The Y axis reports the size of the resulting complete model, expressed in number of particles. The minimum figure is 1 million units to reach over 1 billion particles.

Based on Enoval gearbox strategy change before outlined, two diagrams are shown for comparison. The former is referred to the Full Geometry solution, where the whole set of model's geometrical dimensions have been maintained equal to reality, while the latter one, the so called Axially Reduced Model, includes a marked reduction in axial length in order to limit the entire volume and break it down into smaller particles that will produce a closer-to-reality simulation. Modelling results are summarized in Figure 9 reporting:

- *NpartOil*: Only-oil simulation model (light blue color)
- *Npartic2Phase*: mixture of air+oil simulation model (green color)
- *Ncelle*: cartesian grid for Only-oil simulation (blue color)
- *Max number of cells*: cartesian grid maximum value, not to be trespassed. It is a mathematical limit depending on how the nanoFluidX code has been conceived. Upper limit is resulting $2{,}147*10^9$.

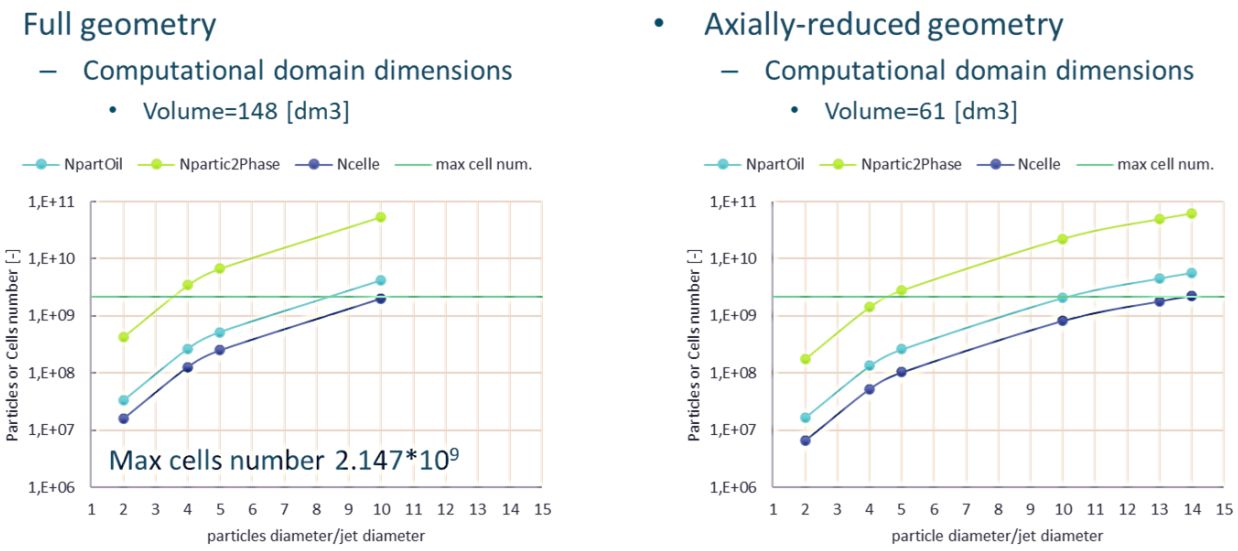

Figure 9 Pre-processing investigation carried out to prepare Enoval model

The objective is to understand the value of the expected amount of oil particles when setting up the ratio "particles diameter/oil jet diameter":

- Minimum Ratio value must not be lower than 5 [-]. This criterium was the result of previous simpler investigations carried out on single and double wheels. In case of real geometry, the corresponding model for oil simulation should raise up to 300 Millions of particles while, in case of axially reduced geometry, the model should be reduced and involve only 100 Millions of particles.
- Desired Ratio value should be around 7 [-]. This will allow to get a good physical feedback on how the flow is evolving inside the gearbox. In case of real geometry simulation, the oil-only model should raise up to 800-900 Million of particles while, in case of axially reduced geometry, the model should go down to 400 Million of particles. Differences are marked and strongly impacting on hardware resources to be used and computational time.

Main conclusion is the following one: reproducing a model of at least 400 Million of particles, or even more, seems to be feasible, but the involved GPU hardware should raise to a number much bigger than 16, that is the number of cards currently used (based on Barbora or DGX-2 cluster availability).

A large number of GPUs, up to 32 or 64, needs to be used for putting this kind of computation into reality.

3.2.2 Deployment of nanoFluidX on multiple GPU-accelerated nodes

The improvement of the main computational phase that the Rotating parts use case is based on requires the deployment of the application solver over multiple GPUs. More specifically, the nanoFluidX run configuration was extended to execute the application solver by leveraging GPUs on multiple nodes. It was tested with two GPU nodes on the IT4Innovations' Barbora cluster. The new configuration "-l select=2:ncpus=24:mpiprocs=4" launches four MPI processes per GPU node, and two GPU nodes in total. It uses one GPU per MPI process, as every GPU node has four NVIDIA V100-SXM2 GPUs. An additional flag was added to the nanoFluidX configuration (filesys_topo SHARED) to enable shared file systems as available on the Barbora nodes.

4 SUMMARY

In this deliverable the optimization tasks that have been performed from a mixed HW/SW integration perspective to deploy the two considered LEXIS Aeronautics use cases that have been presented.

In detail, the main outcomes here illustrated with reference to the optimization of HW and SW resources used in Turbomachinery use case show that the computational phase, the simulation data visualization, and the I/O operations of the underlying application workflow have been successfully enhanced. On one hand, starting from the Turbomachinery use case, the main outcomes here illustrated in terms of optimization of the adopted HW and SW resources show relevant enhancements in the computational phase, in the simulation data visualization, and in the I/O operations of the underlying application workflow. This allowed to improve the execution of the related computational jobs. More specifically, specific software development activities have allowed to move the execution of TRAF solver, that is the main application code used in the Turbomachinery use case, from a solely CPU-based release to a GPU-enabled one with very promising results. Besides that, a preliminary analysis has been started to understand benefits and limitations from a new rewriting of the above-mentioned code focused to leverage the execution acceleration capabilities through FPGA. Moreover, the other research activities carried out in the Turbomachinery use case have been oriented from one side to optimize the implementation of the data visualization jobs included in the considered CFD simulations, and from the other side to analyse how to enhance the write and read operations of TRAF code.

On the other hand, in regard to the Rotating parts use case, the description here provided for the pursued optimization activities underlines that the pre-processing phase and the computational phase of the related application workflow have been so far successfully optimized to pave the way to a more proper and efficient execution of the involved computational tasks. Particularly, to optimize the digital technology implementation of the pre-processing phase, the update of GPU drivers and the identification of a couple of solutions to accelerate the launch stage of the considered application software has become necessary. Furthermore, well-structured research activities have been carried out aiming to identify the most suitable simulation jobs parameters allowing to make as accurate as possible the predictions of the physical phenomena under consideration. Finally, the deployment enhancement of nanoFluidX, that is main application solver used in the Rotating parts use case, has been improved allowing its execution on multiple GPUs not only installed in a single HPC node but also on multiple nodes.

The next steps will be focused on further enhancement of the coupling of HW/SW technologies used in LEXIS to allow the validation the LEXIS Platform by the full execution of the entire application workflows that the two Aeronautics use cases here discussed rely on. Particularly, in the context of the Turbomachinery use case, other HW and SW optimization will be investigated to further improve the obtained results in terms of application solver execution speedup and to allow the entire execution of the considered application workflow, including the writing of the final solution for the long-lasting simulation jobs planned here.

Regarding instead the Rotating parts use case, next research tasks will be focused on industrial-like simulations by identifying the specifically needed HW and SW requirements to implement advanced multiphase approach in the planetary gearbox study and to support closer-to-reality simulations.

While the focus of this deliverable has been on showing improvements in the Aeronautics Pilot and its integration with the LEXIS platform, WP5 has contributed important requirements and design suggestions to the LEXIS co-design process. The result is a platform with secure data handling (encryption, broad security measures) and processing according to industrial standards. Within the remaining months of the project, in this spirit we are confident to trigger and implement further steps in optimisation of the platform and our Pilot.

REFERENCES

[1] LEXIS Deliverable, *D4.5 Definition of Mechanisms for Securing Federated Infrastructures.*

[2] LEXIS Deliverable, *D4.7 Centralized AAI: Coverage of All Significant Systems.*

[3] LEXIS Deliverable, *D3.5 LEXIS Data System Core (Infrastructure).*

[4] "HEAppE Middleware," [Online]. Available: https://heappe.eu.

[5] "ImageMagick®," [Online]. Available: https://imagemagick.org/index.php.

[6] "gnuplot," [Online]. Available: http://www.gnuplot.info/.

[7] "ParaView®," [Online]. Available: https://www.paraview.org/.

[8] "LRZ Best Practices," [Online]. Available: https://doku.lrz.de/display/PUBLIC/Best+Practices%2C+Hints+and+Optimizations+for+IO.

[9] "Altair SimLab," [Online]. Available: https://www.altair.com/simlab/.

[10] "Altair HyperWorks Suite," [Online]. Available: https://www.altair.com/hyperworks.

[11] "Virtual GL," [Online]. Available: https://www.virtualgl.org.

[12] "Turbo VNC," [Online]. Available: https://www.turbovnc.org.