

HORIZON-EUROHPC-JU-2021-COE-01

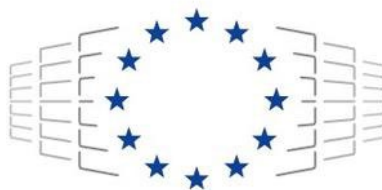


**The European Centre of Excellence for Engineering
Applications**

Project Number: 101092621

D2.5

Updated Report on the CODA Application Use Case



The EXCELLERAT P2 project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101092621. The JU receives support from the European Union’s Horizon Europe research and innovation programme and Germany, Italy, Slovenia, Spain, Sweden and France.

Work Package:	2	Updated Report on the CODA Application Use Case
Author(s):	Michael Wagner	DLR
Approved by	Executive Centre Management	20.05.2025
Reviewer	Damien Dosimont	BSC
Reviewer	Adam Peplinski	KTH
Dissemination Level	Public	

Date	Author	Comments	Version	Status
2025-04-25	Michael Wagner	Initial draft	V0.1	Draft
2025-05-13	Michael Wagner	Updated draft after first review round	V0.3	Draft
2025-05-20	Michael Wagner	Final version	V1.0	Final
2025-12-04	Michael Wagner	Revised version for resubmission	V2.0	Final

List of abbreviations

CFD	Computational Fluid Dynamics
CPU	Central Processing Unit
CRM	Common Research Model
FSDM	FlowSimulator DataManager
GASPI	Global Address Space Programming Interface
GMRES	Generalised Minimal Residual Method
GPI-2	Global Address Space Programming Interface - 2nd generation
GPU	Graphics Processing Unit
HPC	High-Performance Computing
MPI	Message Passing Interface
NASA	National Aeronautics and Space Administration
RANS	Reynolds-Averaged Navier-Stokes
SAneg	Spalart-Allmaras one-equation turbulence model in its negative form
SIMD	Single Instruction, Multiple Data
UC	Use Case

Executive Summary

This document presents the progress made in the CODA Application Use Case UC-1 during reporting period 2, which covers months 13 to 30 (January 2024 - June 2025)¹ of the EXCELLERAT P2 project. Based on the detailed roadmap of the workflow development defined in deliverable D2.1 “Use-Case Execution Roadmap” [7], the workflow of the use case is summarised and the achieved progress with respect to the defined workflow, objectives and success criteria is presented.

In summary, the workflow development for UC-1 has progressed according to the schedule defined in deliverable D2.1. Work has been performed on the individual tasks planned for month 13 to 30 of the project.

¹ As the internal review process of this deliverable started beginning of May 2025, the report covers only the work actually done until M29 (end of May 2025) and includes the expectation of work done until end of June 2025.

Table of Contents

1	Use Case Introduction	7
2	Objectives of the Use Case.....	7
3	Workflow Description.....	8
4	Progress Achieved Since M12	9
5	Next Steps	12
6	References	13

Table of Figures

Figure 1: Visualisation of an exemplary use case simulation: aircraft configuration with mesh (left) and airflow around wing and fuselage (right); both with air pressure as colour gradient.....	7
Figure 2: Baseline vs. improved strong and weak scalability of CODA with UC-1.	10
Figure 3: Speedup and scalability of CODA/Spliss with UC-1 using mixed-precision.	11

1 Use Case Introduction

Computational Fluid Dynamics (CFD) simulations are an increasingly important part of aircraft design. They reduce cost and time of aircraft development and accelerate the introduction of progressive technologies and improvements. Moreover, high-precision CFD simulations are inevitable for assessing future aircraft designs by providing reliable insight into new aircraft technologies and reaching the best overall aircraft performance. They allow to design quieter, safer, and more fuel-efficient planes.

This use case simulates steady airflow at transonic speed and computes typical characteristics such as air velocity and direction, pressure and turbulence (Figure 1). For the use case, the CFD software CODA (CFD by Onera, DLR, and Airbus), in conjunction with the FlowSimulator framework, solves the Reynolds-averaged Navier-Stokes (RANS) equations with a Spalart-Allmaras one-equation turbulence model in its negative form (SAneg). It uses a second-order finite-volume spatial discretisation with an implicit Euler time integration.

The inputs of the use case are aircraft geometries for commercial aviation in the form of unstructured meshes with varying sizes ranging from tens of millions to several billions of elements; typically, with six degrees of freedom per element (ranging up to several hundreds). The aircraft geometries include public models such as the NASA Common Research Model (CRM) and internal models from DLR and Airbus.

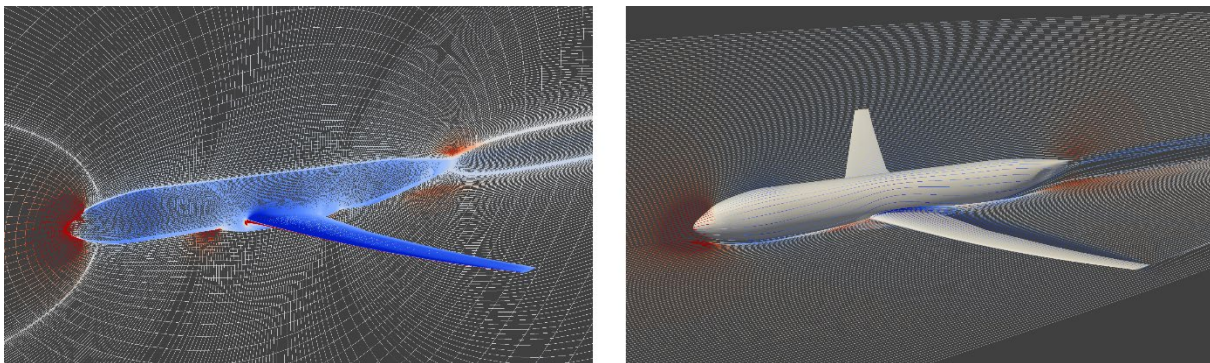


Figure 1: Visualisation of an exemplary use case simulation: aircraft configuration with mesh (left) and airflow around wing and fuselage (right); both with air pressure as colour gradient.

2 Objectives of the Use Case

The objective of the use case is to evaluate and improve the CODA CFD software and the FlowSimulator framework with respect to node and system-level performance and increase their scalability. EXCELLERAT P2 provides the opportunity for early access to pre- and near-exascale architectures and allows to verify our extrapolations from previous strong-scaling tests, which indicated pre-exascale readiness for CODA and FlowSimulator. Real near-exascale testing will show which further performance optimisation is necessary to reach this goal.

Planned external aircraft aerodynamics simulations will require large-scale runs in the range of pre-exascale systems for research and production as well as large sets of medium-scale runs for aerodynamics data production (ensemble runs). Therefore, with the help of the use case, we strive to evaluate and extend the scalability of CODA and FlowSimulator to enable research and production runs at new levels as well as further improve the efficiency of both to allow for aerodynamics data production at medium-scale with high efficiency to reduce time-to-solution and energy consumption on the High-Performance Computing (HPC) systems.

3 Workflow Description

The workflow for the use case consists of three software components: the CFD software CODA, the FlowSimulator framework and the sparse linear systems solver Spliss.

CODA is the CFD software being developed as part of a collaboration between the French Aerospace Lab ONERA, the German Aerospace Center (DLR), Airbus, and their European research partners [2][3]. CODA is a next-generation CFD solver for aerodynamic simulations of fully equipped aircraft and is set to replace its predecessor, the TAU CFD package [1], in production in the European aircraft industry and research organisations. Today, TAU has been in production in the European aircraft industry, research organisations and academia for more than 20 years and was, for instance, used for the Airbus A380 and A350 wing design.

The CODA CFD software sets its focus on, first, a flexible and comprehensive parallelisation concept suited for current and future HPC systems and, second, on algorithmic efficiency using strong implicit solvers, higher-order spatial discretisation via the Discontinuous Galerkin method featuring hp-adaptation in addition to finite volumes with maximum code share, and seamless integration into Python-based multi-disciplinary process chains via FlowSimulator. For that, CODA uses classical domain decomposition to utilise distributed-memory parallelism via Message Passing Interface (MPI) and, additionally, the Global Address Space Programming Interface (GASPI) implementation GPI-2 as an alternative to MPI, which allows for efficient one-sided communication to reduce network traffic and latency. For both, CODA supports overlapping of halo-data communication with computation to hide network latency and further increase scalability. Besides classical domain decomposition, CODA employs a hybrid two-level parallelisation to utilise shared-memory parallelism for multi- and many-core architectures. CODA implements sub-domain decomposition, where each domain is further partitioned into sub-domains, each of which is processed by a dedicated software thread that is mapped one-to-one to a hardware thread to maximise data locality. The hybrid approach allows utilising all layers of parallelism and providing a flexible adaptation to different hardware architectures [4].

The CODA CFD software is a part of the multi-disciplinary analysis (MDA) framework FlowSimulator [5], which provides plug-ins for all steps of a full aircraft simulation as well as a seamless integration into multi-disciplinary simulations. In particular, CODA uses FlowSimulator's core component, the FlowSimulator DataManager (FSDM) for I/O, where various I/O libraries are supported. FSDM is an open-source software hosted by DLR. FSDM provides the FSMesh class, which is the preferred container for the exchange of data among FlowSimulator plug-ins. FSDM is MPI parallelised and an FSMesh instance is a distributed representation of the data, usually containing information on the geometry, the (computational) mesh, flow fields/solutions, and coupling strategies.

Furthermore, CODA employs the sparse linear system solver Spliss [6]. Spliss is a high-performance, hardware-aware sparse linear system solver designed specifically for computational fluid dynamics (CFD) applications. It targets the efficient solution of large, sparse, and often symmetric indefinite systems that arise in implicit CFD methods, such as those used in finite volume or finite element discretisations. Built with modularity and performance in mind, Spliss is decoupled from the core CFD solver (e.g., CODA), enabling it to be integrated seamlessly while abstracting solver complexity. The solver supports advanced optimizations tailored to modern hardware, including GPU acceleration via CUDA, optimized memory access patterns, and efficient kernel execution for sparse matrix operations. By offloading computationally intensive linear solves to GPUs without requiring changes to the CFD codebase, Spliss significantly improves performance and scalability, making it well-suited for large-scale, high-fidelity simulations where solution speed and resource efficiency are critical.

At its core, Spliss is built around the recognition that the performance of CFD simulations is often bottlenecked by the linear solver phase, especially in high-resolution, three-dimensional simulations involving complex geometries and turbulent flow regimes. To address this, Spliss is engineered not only for numerical robustness and convergence speed but also for deep integration with heterogeneous computing architectures, particularly Graphics Processing Units (GPUs). Unlike traditional solvers that are tightly coupled to specific CFD frameworks, Spliss maintains a clean separation from the application code, such as CODA, by exposing a well-defined, low-level interface. This decoupling allows CFD developers to leverage advanced solver capabilities without modifying their core simulation logic, promoting code reusability and maintainability.

One of Spliss's key strengths lies in its ability to exploit hardware-specific optimizations without burdening the user with low-level implementation details. For instance, it employs custom CUDA kernels optimized for sparse matrix. It also supports dynamic load balancing across GPU streaming multiprocessors and efficient use of shared memory to reduce redundant data access. Furthermore, Spliss integrates advanced preconditioning techniques, such as incomplete LU factorization (ILU) and algebraic multigrid (AMG), that are adapted to the specific structure of CFD matrices, improving convergence rates for challenging problems like high-Reynolds-number flows.

The solver supports both iterative methods (e.g., GMRES, BiCGSTAB, and CG for symmetric systems) and, in certain configurations, direct solvers for smaller or critical subsystems. Its modular design allows for easy configuration of solver parameters, preconditioners, convergence criteria, and hardware backends, enabling users to tailor performance to their specific problem class and available hardware. Additionally, Spliss includes comprehensive profiling and diagnostic tools that provide insights into solver behaviour, such as iteration counts, residual evolution, and GPU kernel execution times, facilitating performance tuning and debugging.

Another significant advantage of Spliss is its portability and extensibility. While initially developed for Nvidia GPUs, its architecture supports future expansion to other accelerators, such as AMD GPUs, through abstraction layers and a plugin-based design. This forward-looking approach ensures long-term relevance in the rapidly evolving landscape of high-performance computing.

4 Progress Achieved Since M12

During the reporting period, five main tasks were carried out: First, we assessed the improved scalability and compared it to the baseline scalability of CODA and FlowSimulator on the largest available partition of DLR's main production system CARA with the NASA common research model in a strong and weak-scaling scenario. Second, we evaluated the newly introduced mixed-precision mode in the sparse linear solver Spliss. Third, we evaluated the performance and scalability of CODA, FlowSimulator and Spliss delivered in container images on the DLR HPC systems and compared them to the installations using the native software stack of the system. Fourth, we compared the performance of CODA on various upcoming Central Processing Unit (CPU) and GPU architectures. Fifth, we evaluated the newly developed hierarchical mesh partition method in FlowSimulator.

Please note that the actual code improvements were developed outside the scope of this project. The use case and the work within EXCELLERAT P2 focus on the evaluation and demonstration of this improvements on current state-of-the-art HPC systems.

Evaluation of Improved Scalability of CODA and FlowSimulator

First, we focused on evaluating the scalability of CODA on CARA with Use Case UC-1. This use case solves the Reynolds-averaged Navier-Stokes equations with a Spalart-Allmaras turbulence model in its negative form (SA-neg). The use case runs on an unstructured mesh from the NASA Common Research Model (CRM) with about 5 million points and 24 million volume elements. This relatively small mesh was chosen for a strong-scalability analysis of CODA. Production meshes are typically at least ten times larger and accordingly achieve comparable efficiency on much higher scales. For the weak scalability analysis, we use different mesh sizes from the CRM mesh family ranging from 3 to 192 million elements and solve the use case with an according number of cores.

Figure 2 highlights the scalability improvements of CODA on the CARA HPC system based on the AMD Naples CPU architecture. CODA achieves about 83% parallel efficiency (vs. 61% baseline) on the largest available partition on CARA with 512 nodes and 32,768 cores in the strong-scaling scenario. In the weak-scaling scenario, a parallel efficiency of 96% (vs. 72% baseline) was achieved on 32,768 cores. Numerical accuracy remained unchanged across all measurements.

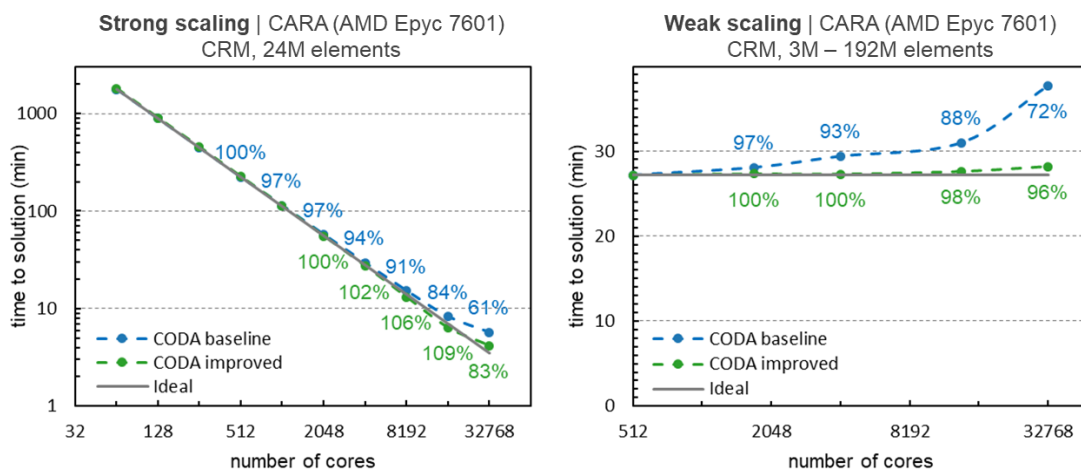


Figure 2: Baseline vs. improved strong and weak scalability of CODA with UC-1.

The introduction of new features in CODA and FlowSimulator is accompanied by comprehensive numerical and computational performance analysis, providing clear insight into efficiency, accuracy, and scalability; whereas the figure above just highlights the baseline and current scalability. The publicly accessible use case serves as a consistent reference for external validation and community testing, enabling transparent sharing and reproducibility. It offers valuable feedback for refining algorithms and improving performance across diverse hardware. It also supports early access testing and porting efforts on emerging CPU and GPU architectures, ensuring compatibility and optimal performance from the outset. Furthermore, the use case is used as a robust baseline for scalability regression testing in every internal release, helping to detect performance regressions early and maintain high computational efficiency across all software updates.

Evaluation of Mixed-precision in the Linear Solver Spliss

In addition to improvements in scalability, optimisations in compute speed were also considered. Among other improvements, we evaluated the use of mixed-precision floating-point calculations in the linear solver Spliss. A typical solver stack in Spliss that is used by CODA is, for example, the Generalised Minimal Residual Method (GMRES) with Jacobi preconditioning. In mixed-precision mode, the inner loops (Jacobi preconditioner) of the linear solver are computed with single-precision floating-point arithmetic (32 bits), whereas the outer loops (GMRES) still use double-precision floating-point arithmetic (64 bits). The advantages are, on the one hand, the utilisation of twice the number of entries per Single Instruction, Multiple Data (SIMD) instruction (for compute-bound sections) and the halving of the amount of data to be loaded from the memory (for memory-bound sections). Consequently, the CODA's calculation time could be accelerated by up to 72% (see Figure 3). The extent of the acceleration depends on the test case and the ratio of inner to outer loops. On average, users report an acceleration of around 30% for the entire simulation.

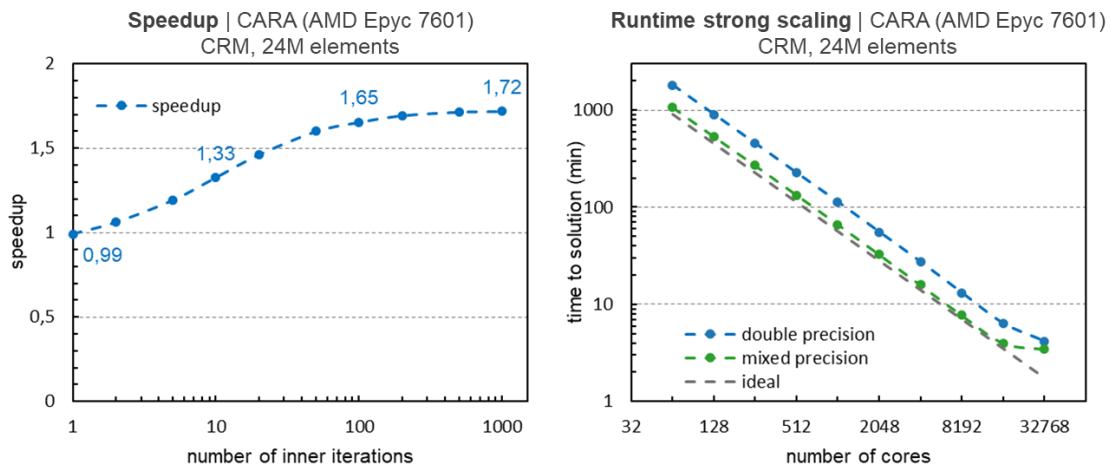


Figure 3: Speedup and scalability of CODA/Spliss with UC-1 using mixed-precision.

Containerised Workflow Deployment

CODA, FlowSimulator, Spliss and all the workflow dependencies can now be delivered as a single container image. Thus, CODA can be executed in a container on any HPC system independently of the installed software stack. The main benefits are easier delivery to users and customers, simpler deployment on different systems and significantly improved portability.

We evaluated the performance and scalability of the container images on the DLR HPC systems where we can compare them to the installation using the native software stack of the system. The measurements showed comparable performance and scalability for the containerised workflow. Thus, the delivery and deployment of the containerised workflow will become the default for the next internal releases.

Comparison of CODA's Performance on Upcoming CPU and GPU Architectures

As part of our ongoing efforts to test and evaluate CODA and FlowSimulator on new CPU and GPU architectures, we have so far evaluated the following systems (entries in brackets are from the previous reporting period):

- AMD: (Zen1, Zen2, Zen3, Zen4)
- Intel: Sapphire Rapids, (Icelake)
- ARM-based: Graviton 4, Nvidia Grace, (Graviton2, Graviton 3)
- GPU: Nvidia H100, AMD Mi210, (Nvidia A100)

For the evaluation, we use standardised benchmarks and a containerised version of CODA and FlowSimulator, including the use case UC-1. These measurements allow us to adapt CODA to new architectures during the early-access phase and evaluate which systems offer best performance ahead of deployment to new full-scale HPC systems as well as provide valuable insight for designing DLR's own future HPC systems.

Evaluation of Improved Mesh Partitioning in FlowSimulator

Following our investigation into improvements to mesh partitioning in the previous reporting period, we have now evaluated a newly developed hierarchical partitioning method. The hierarchical partitioning method distributes the mesh at three levels: first, it distributes the mesh across all involved compute nodes, then within each node across all MPI processes, and finally for each MPI process across all threads. This method is flexible to use different graph partitioners such as Parmetis or Zoltan for each level. It significantly speeds up mesh partitioning for large meshes and large numbers of cores (by up to one order of magnitude), while not degrading the resulting load balance for the CFD solver. The improved mesh partitioning also allows simulations of larger meshes with more than one billion elements and we are now able to successfully run simulations on 131,072 cores (the full DLR CARO system).

5 Next Steps

According to the schedule defined in Deliverable D2.1, we continue with three tasks: First, the continuous analysis of CODA and FlowSimulator efficiency and capability improvements and, second, the continuous improvement of CODA and FlowSimulator efficiency and capabilities. Finally, we will start working on the CODA and FlowSimulator efficiency demonstrator.

The next steps are, first, the continuous evaluation of CODA's and FlowSimulator's scalability with UC-1, in particular, for new features and improvements. This includes scalability regression tests for new internal releases. Second, the evaluation of CODA and FlowSimulator with UC-1 on further HPC systems via the containerised workflow delivery. Third, we continue the testing and evaluation of upcoming CPU and GPU architectures. Fourth, we continue testing, evaluating and improving all workflow components in FlowSimulator to improve the support for large meshes and large core counts. This will lead to the scalability and efficiency demonstrator for CODA and FlowSimulator.

6 References

- [1] D. Schwamborn, T. Gerhold and R. Heinrich, “The DLR TAU Code: Recent Applications in Research and Industry”, In Proc. of the European Conference on Computational Fluid Dynamics, ECCOMAS CFD, 2006. <https://elib.dlr.de/22421>
- [2] T. Leicht, J. Jägersküpper, D. Vollmer, A. Schwöppe, R. Hartmann, J. Fiedler and T. Schlauch, „DLR-Project Digital-X-Next Generation CFD Solver 'Flucs'“, Deutscher Luft- und Raumfahrtkongress 2016. <https://elib.dlr.de/111205/>
- [3] I. Huisman, S. Fechter, and T. Leicht. "HyperCODA—extension of flow solver CODA towards hypersonic flows." In New Results in Numerical and Experimental Fluid Mechanics XIII: Contributions to the 22nd STAB/DGLR Symposium. Springer International Publishing, 2021. https://doi.org/10.1007/978-3-030-79561-0_10
- [4] M. Wagner, „Scalability Evaluation of the CFD Solver CODA on the AMD Naples Architecture”, In Sustained Simulation Performance, pp. 95-106, 2023. https://doi.org/10.1007/978-3-031-18046-0_7
- [5] M. Meinel and G. Einarsson, “The FlowSimulator Framework for Massively Parallel CFD Applications”, In PARA 2010. <https://elib.dlr.de/67768>
- [6] O. Krzikalla, A. Rempke, A. Bleh, M. Wagner and T. Gerhold, “Spliss: A Sparse Linear System Solver for Transparent Integration of Emerging HPC Technologies into CFD Solvers and Applications”, In New Results in Numerical and Experimental Fluid Mechanics XIII, pp. 635-645, 2021. https://doi.org/10.1007/978-3-030-79561-0_60
- [7] EXCELLERAT P2 project, D2.1 “Use-Case Execution Roadmap”