D2.2
Report on application improvements - II semester

Project Title	dealii-X: an Exascale Framework for Digital Twins of the Human Body
Project Number	101172493
Funding Program	European High-Performance Computing Joint Undertaking
Project start date	1 October 2024
Duration	27 months

dealii-X has received funding from the European High-Performance Computing Joint Undertaking Programme under grant agreement Nº 101172493

Deliverable title	Report on application improvements - II semester
Deliverable number	D2.2
Deliverable version	v1
Date of delivery	August 31, 2025
Actual date of delivery	August 31, 2025
Nature of deliverable	Report
Dissemination level	Public
Work Package	WP2
Partner responsible	POLIMI

Abstract	We report on the progress in the biomedical applications that leverage the high-performance and exascale computing capabilities of deal.II. These include lung modeling (TUM), cardiac modeling (POLIMI), brain modeling (FAU), liver modeling (WIAS), and cell mechanobiology (UNIBS). Additionally, we report on the advancements of the low-code/no-code interface developed by Dualistic/eXact-Lab.
Keywords	Applications; Exascale; Lung; Heart; Brain; Liver; Cell

Document Control Information

Version	Date	Author	Changes Made
0.1	28/08/2025	Michele Bucelli (POLIMI)	Initial draft with info of all partners
0.2	30/08/2025	Michele Bucelli (POLIMI)	Improvements in various places
0.3	31/08/2025	Martin Kronbichler	Minor improvements
1.0	31/08/2025	Michele Bucelli (POLIMI)	Final version

Approval Details

Approved by: Martin Kronbichler
Approval Date: August 31, 2025

Distribution List

- Project Coordinators (PCs)
- Work Package Leaders (WPLs)
- Steering Committee (SC)
- European Commission (EC)

Disclaimer: This project has received funding from the European Union. The views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the European Union or the European High-Performance Computing Joint Undertaking (the “granting authority”). Neither the European Union nor the granting authority can be held responsible for them.

Contents

1 Introduction

In this document, we report on the status of the exascale human applications part of the dealii-X project:

- Lungs (TUM)
- Heart (POLIMI)
- Brain (FAU)
- Liver (FVB-WIAS)
- Mechanobiology (UNIBS)

Additionally, we report on the advancements of the low-code/no-code interface developed by Dualistic/eXact-Lab.

1.1 Purpose of the Document

This report collects the updates from the project partners (TUM, POLIMI, FAU, FVB-WIAS, UNIBS, Dualistic/eXact-Lab) into a single document. Each section reports on advancements on the corresponding application, and identifies the upcoming steps in the project.

1.2 Structure of the Document

- Section 2: Reports on Applications

2 Reports on applications

2.1 Lungs (TUM)

This section reports on the progress of the lung application. The lung application is built on the high-performance library ExaDG, which itself relies heavily on the deal.II matrix-free framework as its engine. ExaDG implements discontinuous Galerkin as well as continuous Galerkin solvers for engineering problems. Although the main focus of ExaDG has been computational fluid dynamics (also successfully applied previously in respiratory mechanics) with discontinuous, hypercube-shaped elements, more recent developments and improvements (Schussnig et al., 2025) have enabled solving structural problems with continuous, simplex-shaped elements, which is more typical and appropriate for the partial differential equations underlying elastodynamics.

In accordance with our goal of simulating the elastodynamic processes in the very fine alveolar structures of the lung parenchyma and understanding how these fine structures behave under the influence of surface tension effects, we are conducting initial simulations with an alveolar geometry using ExaDG. In Figure 1, a mesh of a smaller alveolar geometry that we have available is visible. The figure also shows an exemplary displacement field under axial tension. For such complex, thin-walled geometries, the construction of linear solver and preconditioner combinations constitutes an essential part of the development process. We have been experimenting with different setups to find the best solution. So far, multigrid preconditioning is the most promising approach.

Parallel to our work on using the alveolar geometry for ExaDG simulations, we have also been evaluating the applicability and implementing the enhanced-surface finite element formulation for the surfactant, which we had mentioned in the previous deliverable D2.1. We expect this formulation to be incorporated into ExaDG in the very short term.

2.2 Heart (POLIMI)

This section reports on updates on the heart modeling library lifex, which relies on deal.II to build a comprehensive multiphysics tool for the simulation of the cardiac

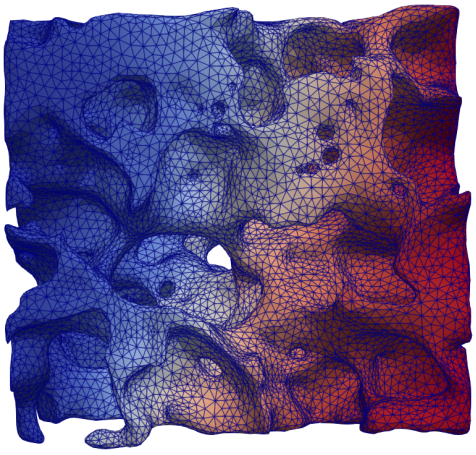

Figure 1: Mesh and displacement field of an alveolar structure under axial tension.

function.

As mentioned in Deliverable 2.1, prior to the start of the dealii-X project lifex relied on conventional matrix-based solvers for all the physical models involved. The solution of linear systems constituted a major bottleneck for performance and parallel scalability (Africa et al., 2023, 2024), posing a limitation to the scale of problems that could be effectively solved with the library.

Therefore, the POLIMI team is working on migrating the lifex solvers to deal.II's matrix-free infrastructure, and is extensively refactoring the code organization towards this goal. In the new implementation, each concrete PDE model (such as cardiac electrophysiology, muscular mechanics or myocardial perfusion) builds upon the same abstract software components (i.e., it is derived from the same classes), as opposed to the old implementation, where different models would often reimplement common tasks (such as the initialization of finite element spaces or linear system assembly). Besides significantly improving the code organization and its maintainability, this centralized structure significantly facilitates the implementation of new physical models and their improvement, and overall provides a structured scaffolding for the future library developments.

The abstract physical model classes at the core of this new implementation leverage deal.II's facilities to support many solver configurations in a flexible way: they allow for scalar- and vector-valued problems, linear and nonlinear, steady and unsteady, on hexahedral and tetrahedral meshes, and support both matrix-based and matrix-free solvers. Finite element evaluation tasks are carried out with the `dealii::FEEvaluation` class, instead of `dealii::FEValues` as done in the previous

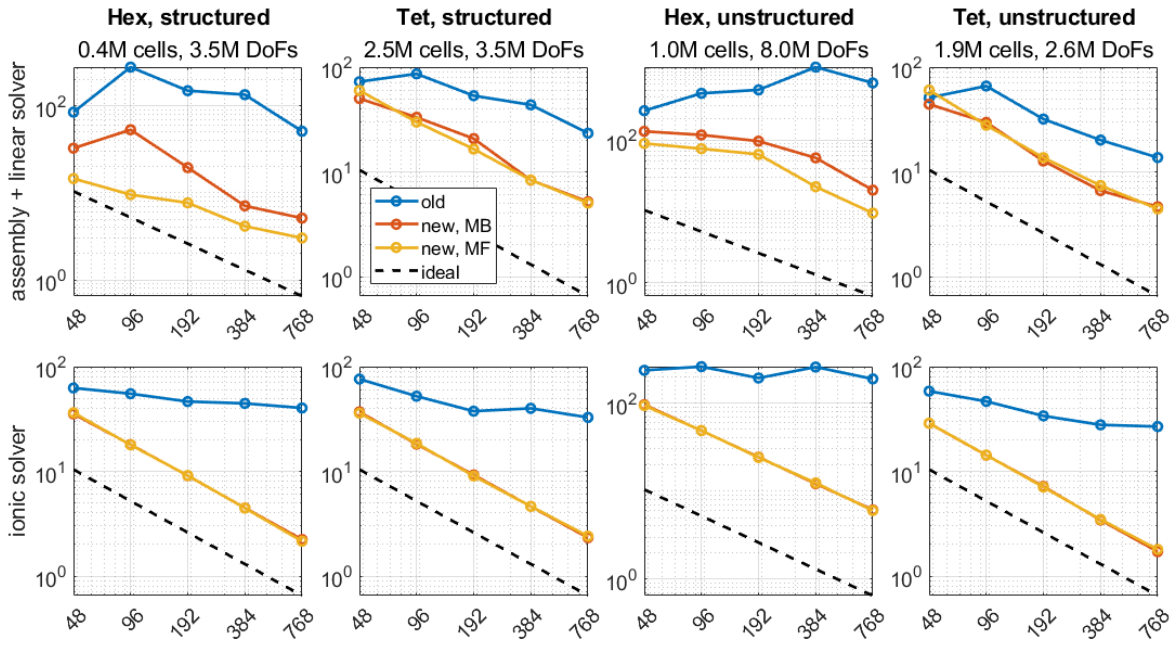

Figure 2: Strong calability of the lifex monodomain solver, in previous (matrix-based) implementation and new (matrix-based and matrix-free) implementation.

implementation. This enables a significant performance improvement already in the matrix-based solvers, thanks to the use of sum factorization, SIMD vectorization, and to improved cache friendliness.

The solver for the monodomain problem of cardiac electrophysiology has been implemented on top of the new infrastructure, and its parallel performance has been compared to the one of the old code by running a strong scalability test based on the Niederer benchmark (Niederer et al., 2011). The results (see Figure 2) show a pronounced improvement in both absolute wall times and parallel scalability, with wall times reduced by a factor 1.5–30, depending on mesh type and number of parallel cores. We note in particular that the vectorization of the ionic solver has lead to a much clearer scalability. This is even more noticeable when observing that the cost of solving ionic models often dominates electrophysiology simulations (Africa et al., 2023).

Additionally, we have migrated to the new implementation of the multicompartment Darcy solver for myocardial perfusion (Zingaro et al., 2023), and the solver for generating rule-based cardiac fiber architectures (Piersanti et al., 2021). Finally, we are in the process of extending this new framework to the cardiac mechanics solver (Fedele et al., 2023), as well as coupling the latter to the Darcy model to simulate the behavior of a porous medium subjected to fluid flow.

The upcoming work will follow along these lines, by progressively extending the new infrastructure to additional models (e.g. active force generation for muscular contraction) and introducing coupling between the existing models to build multi-physics simulations (e.g. electromechanics or poromechanics for myocardial perfusion). Complementary to this, we will proceed by expanding the set of numerical methods supported by the core solvers. Most notably, the code currently supports only black-box preconditioning methods (AMG, ILU, etc.), which are not well suited for the matrix-free solvers. Since preconditioning is crucial for the robustness and efficiency of cardiac mechanics solvers, this will be addressed shortly, by integrating a multigrid method within the new code (Schussnig et al., 2025).

2.3 Brain (FAU)

Here, we report on the progress within the ExaBrain library which is developed to facilitate high resolution full human brain finite element simulations using a complex nonlinear poro-viscoelastic material model.

On the one hand, we transferred our code to the National High Performance Computing Center (NHR@FAU) at FAU in Erlangen. This allowed us to perform first benchmarks on the code performance and scalability in an HPC environment which will also serve as the baseline for upcoming developments. Overall, we compared the performance for several system sizes and two different direct solvers. Figure 3 exemplarily shows the results for the – so far – largest system consisting of 24,576 hexahedral Q2P1 (quadratic in displacements, linear in pore pressure) elements and 644,916 degrees of freedom. The simulations were run on HPC system “Fritz” with our baseline parallel direct solver SuperLU-dist and a newly available MUMPS direct solver, which was incorporated into deal.II through our project partners at University of Pisa and the work of the MUMPS team at INPT. The new MUMPS solver decreases the memory requirement by approximately $50\,\%$ to $75\,\%$, depending on the number of parallel processes (Figure 3, left). A crucial improvement towards larger systems required for the full human brain. The system assembly does not depend on the solver and scales perfectly with the number of processes (Figure 3, middle). While the new MUMPS solver is significantly faster than SuperLU-dist, it does not show good scalability through parallelization (Figure 3, right). (The system could not be solved using the SuperLU-dist solver with 64 MPI ranks as this combination exceeded the memory capacity of 256 GB of one compute node at “Fritz”.)

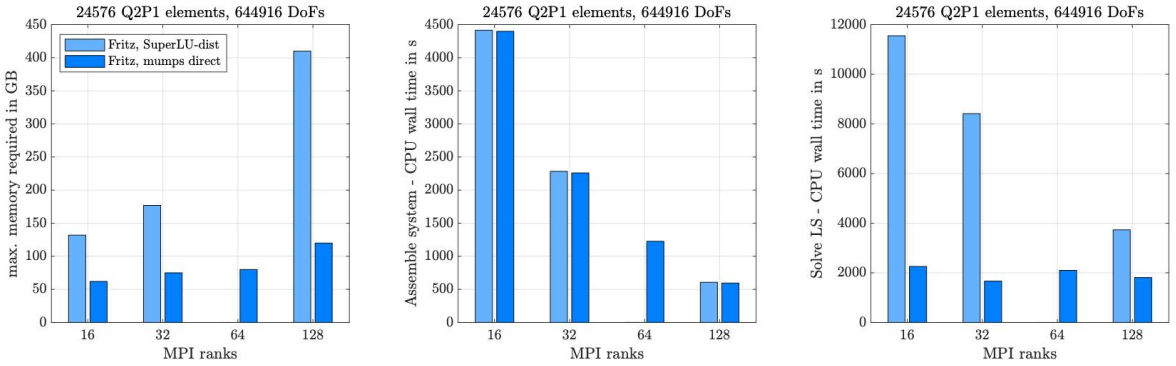

Figure 3: Comparison between two parallel direct solvers (SuperLU-dist and MUMPS): memory usage (left), CPU wall time for system assembly (middle) and CPU wall time so solve the linearized system (right).

Wall time of linear solver - ExaBrain project - 16 MPI Processes, 8 OpenMP Threads

	SUPERLU	Direct MUMPS	BLR-GMRES	BLR reuse in Newton	BLR update only on fail
Medium Q2P1 (DoFs: 116,161)	67.8	23.2	31.8	14.7	38.9
Medium Q3P2 (DoFs: 38,864)	270.7	111.7	141.5	61.0	111.3
Large Q2P1 (DoFs: 84,572)	944.0	374.2	294.8	125.3	189.9
Large Q3P2 (DoFs: 291,708)	9874.0	2507.0	1509.0	627.8	747.7
XLarge Q2P1 (DoFs: 644,916)	Failed*	9647.0	4492.0	1704.8	1745.9

Figure 4: Comparison between CPU wall time (s) required by direct and iterative solvers on different sizes of the linear systems solved in a full simulation. SuperLU - dist failed to reach the end of the simulation in the maximum time allowed on the maximum dimension simulation. Each MPI process is using 8 OpenMP Threads.

Further improvements for the solver phase of the linearized system are possible by employing iterative solvers. We report some first comparisons between the direct solvers previously mentioned and GMRES as an iterative solver in Figure 4. The simulations were run on the University of Pisa HPC system “Toeplitz”. GMRES is preconditioned with the MUMPS solver using the mixed precision Block Low Rank feature. Multiple strategies can be employed to precondition GMRES with BLR: the same preconditioner can be reused to precondition multiple executions of the GMRES solver reducing the number of times the preconditioner has to be evaluated and thus reducing the overall time spent in the linear solver. In Figure 4 we show the differences between evaluating the preconditioner every time we have to solve a linear system (i.e., at each Newton-Raphson step), evaluating the preconditioner once at the beginning of each Newton-Raphson iteration, and updating the preconditioner only if the GMRES iterative solver fails to converge. Ongoing work is focusing on optimizing the number of times the preconditioner must be evaluated in relation to the time required for the convergence of GMRES to further increase the speed of the solve phase.

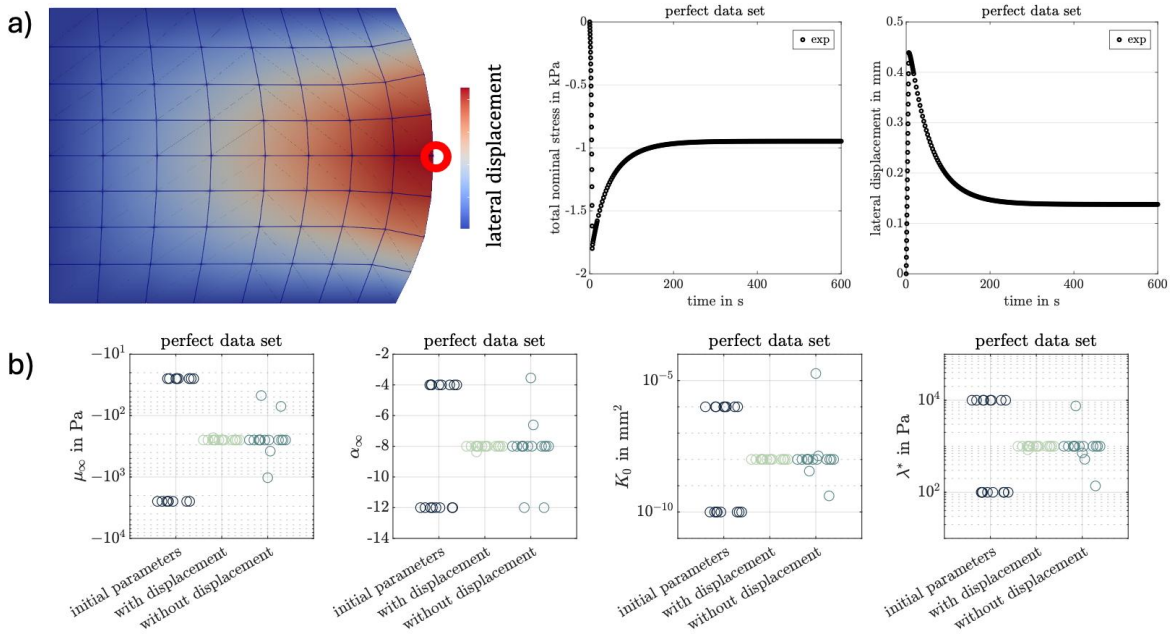

Figure 5: Incorporation of additional information (here the lateral displacement) improves reliability of the inverse parameter identification for relaxation tests using a poroelastic material model.

On the other hand, as full brain simulations require proper material parameters, FAU is working on the efficiency and reliability of the inverse parameter identification. Here, our new HPC possibilities allow us to perform numerical experiments that can then help to identify potential experimental and algorithmic improvements. Figure 5 shows the results of such a numerical experiment. Starting with a known set of poroelastic parameters ($\mu_\infty = 250\,\mathrm{Pa}$, $\alpha_\infty = 8$, $K_0 = 10^{-8}\,\mathrm{mm}^2$ and $\lambda^* = 103\,\mathrm{Pa}$), we performed a relaxation experiment of a cylindrical specimen and evaluated the total nominal stress as well as the lateral displacement of a point on the lateral surface at half of the specimen height (see Figure 5a). We then used this numerical "experimental" data as an input for our inverse parameter identification with 16 different sets of initial parameters and included once only the total nominal stress into the objective function and once additionally the lateral displacement (equally weighted). Even for this simplified poroelastic example using "perfect" experimental data, the nominal stress response alone does not provide enough information to identify a unique parameter set. In fact, only in 12 out of 16 cases we recover the correct parameter set. The situation improves significantly if we incorporate the lateral displacement as additional information (see Figure 5b).

For our poro-viscoelastic human brain model, the number of unknown model parameters increases from four to seven strongly coupled material parameters. There-

fore, numerical experiments as shown in Figure 5 gain even more importance in understanding the model response and interpreting the results from inverse parameter identifications. With an increasing number of model parameters, we have to cover a significantly larger parameter space where we will benefit substantially from our improvements on the linear solver and the inverse identification itself.

2.4 Liver (FVB-WIAS)

This section reports on the liver application. The computational model is built on deal.ii and on the implementation of reduced Lagrange multipliers for multidimensional PDEs. An efficient approach for handling the multiple scales of the problem (fluid flow in the vasculature and elastodynamics in the tissue) is necessary to handle large scale problems.

The reduced Lagrange multipliers approach (Heltai and Zunino, 2023) formulates the weak coupled (fluid-structure) problem in a weak Lagrange multiplier fashion, applying then a model order reduction based on the properties of the relevant physics (i.e., the approximation that flow can be described by one-dimensional Navier-Stokes equations). The work in WP2.4 focused on the application of this framework to multiscale elasticity.

The reduced Lagrange multiplier model extends the previous multiscale model proposed in Heltai and Caiazzo (2019) to the case of Dirichlet boundary conditions enforced at the vessel-tissue boundaries. This condition is necessary to handle the coupling with pulsatile vasculature. To this purpose, a non-standard interface condition has been considered, which removes macroscopic motion (translational and rotational) and only enforces the coupling at the level of local normal displacement of the vessel. The mathematical framework of Heltai and Zunino (2023) provides the right structure for selecting the proper reduced-order space, and extending these results we proved well-posedness at the continuous level of the new formulation. The model has been used to perform in silico experiments relating the properties of the microstructure to the effective elastic parameters of a tissue sample. These experiments represent the first step towards the formulation of an inverse problem for the parametrization of effective liver samples. Results have been submitted for publication to the International Journal of Numerical Method for Biomedical Engineering.

Following the development of the model, current work is considering the coupling

with an active one-dimensional model for the blood flow, extending the previous model of Heltai et al. (2021). The coupling is implemented representing the one-dimensional network (described by nodes and edges) within the three-dimensional domain as a discrete set of singular points, on which a normal displacement is imposed on the tissue, based on the vessel pulsation. The resulting tissue pressure is, in turn, considered as external force in the blood flow model. The coupling has been first implemented in a staggered fashion, performing preliminary investigations on the stability of the model. A monolithic implementation is under development.

The upcoming work will focus on the generation of realistic vasculatures using physiological parameters specific for the liver, as well as the connection with data assimilation methods to integrate available data for model parametrization.

2.5 Mechanobiology (UNIBS)

This section presents recent advancements in mechanobiology, led by the University of Brescia (UNIBS), focusing on the development of advanced computational models to simulate cell motility. Understanding—and ultimately controlling—cell motility is a scientific challenge of the highest order, which underpins metastasis, embryogenesis, and hemostasis. They all rely on non-exiting force-bearing networks, which are purposefully assembled and dismantled in response to physiological demands. Central to this research is the mechanobiological interplay between actin cytoskeleton dynamics and receptor-mediated interactions between the cell membrane and the extracellular matrix. Traditional continuum theories, such as those proposed by Larché and Cahn and Mixture Theories, fall short in capturing the complex coupling of mechanical, chemical, and transport phenomena that govern cytoskeletal reorganization, as they lack the ability to describe the dynamic creation and disassembly of networks. The mathematical foundations of our approach—including the underlying kinematics and balance equations—are detailed in an upcoming publication (Salvadori et al., 2025). The computational model is implemented in dealii-X and employs staggered algorithms to solve the coupled PDE system involving chemo-transport and mechanics. Manufactured solutions have been tested in parallel computing environments utilizing multiple processors and distributed memory architectures.

This deliverable focuses on a Lagrangian approach, adopting both global and adaptive local mesh refinement strategies, with the latter guided by the KellyEr-

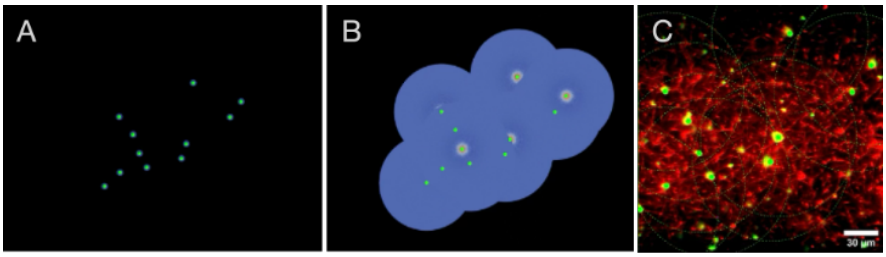

Figure 6: The dealii-X simulation of the formation of load-bearing networks.

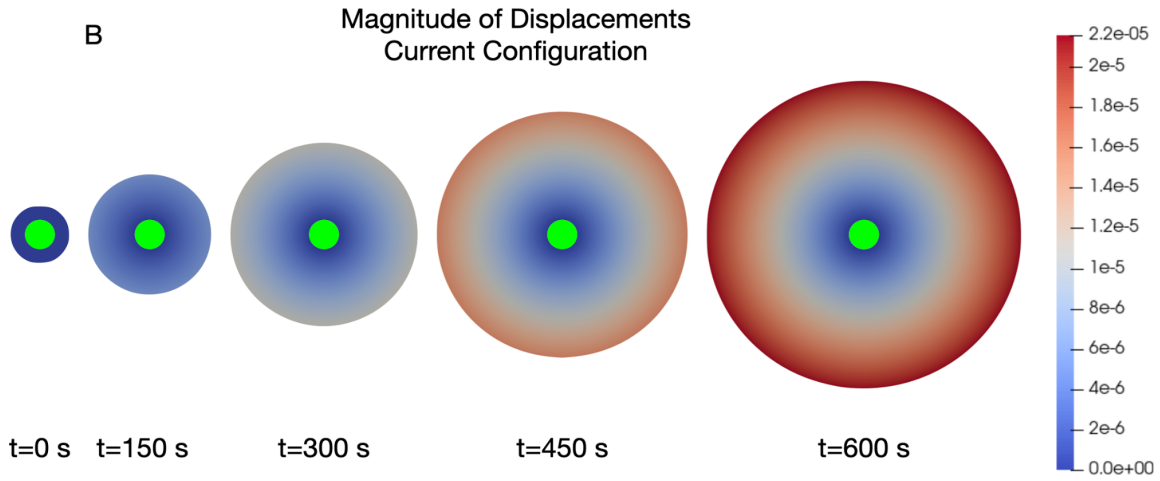

Figure 7: The dealii-X simulation of the formation of load-bearing networks.

rorEstimator in deal.II to estimate cell-wise error and improve solution accuracy. Simulations show the ability of the code to generate force-bearing domains upon polymerization of monomers. These simulations apply well in hemostasis, where the growing domain simulates the development of fibrin networks from activated platelets with real data.

Figure 6A shows the initial location of platelets, experimentally identified, surrounded by thrombin that catalyzes the clot formation. Figure 6B shows the dealii-X simulation of the load-bearing fibrin network at the end of the simulation, to be compared with the experimental data collected Figure 6C. A case study simulation moved from an initial tessellation made of 432 quadrilateral elements, distributed across 8 processors using parallel memory architectures. Adaptive mesh refinements were applied every 150 seconds during a virtual experiment lasting 10 minutes. The number of elements increased progressively from 432 to 6249 by $t = 600\,\mathrm{s}$, passing through 819 ($t = 150\,\mathrm{s}$), 1506 ($t = 300\,\mathrm{s}$), and 3264 ($t = 450\,\mathrm{s}$). Upon polymerization, the volume of the force-bearing network increases with time. Constraining displacements at the platelet boundary, the volumetric expansion of the fibrin

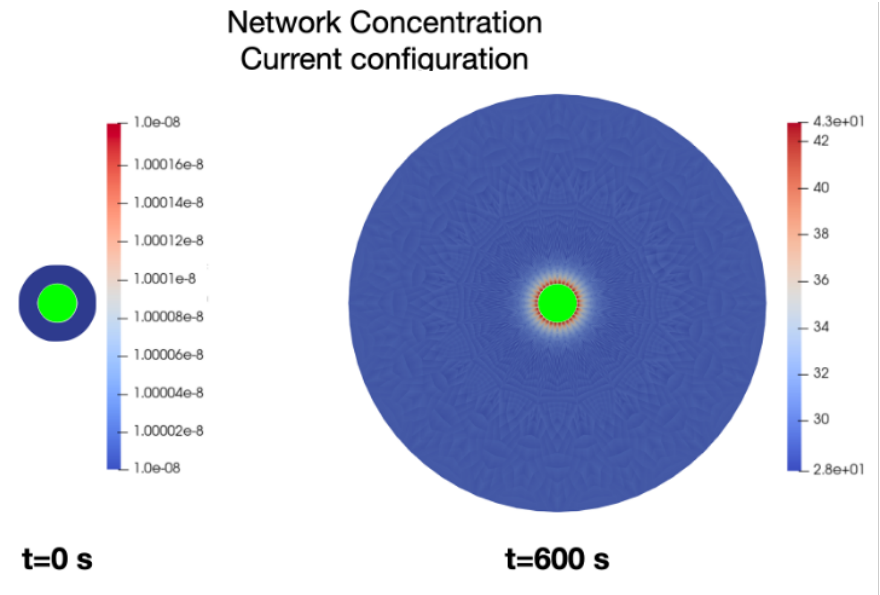

Figure 8: The dealii-X simulation of the formation of load-bearing networks.

network in the current configuration of the domain at five different time instants ($t = 0, 150, 300, 450$, and $600\,\mathrm{s}$) is plotted in Figure 7.

The network concentration is reported in Figure 8. Since the activation signal is applied within a fixed region surrounding the platelet in the current configuration, we observe a higher fibrin concentration near the platelet, with decreasing concentrations farther from the activation region.

A Eulerian framework for solids suits well the goal of the dynamic creation and disassembly of networks. The Eulerian perspective is particularly effective in cellular motility, where signaling occurs in localized zones naturally described in the current configuration. Future work will focus on extending the current code, providing realistic mechanisms for the generation and development of the actin cytoskeleton using parameters specific to fibroblasts and potentially other cell types.

2.6 Dualistic/eXact-Lab

2.6.1 Introduction

The following section reports the progress on the Low-code/No-code interface as well as the related backend. The development follows the approach and the design

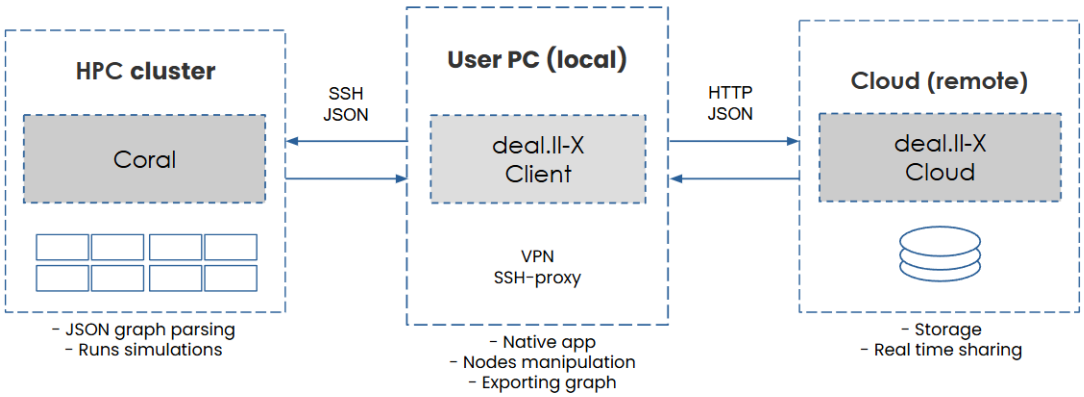

Figure 9: Architecture of the Low-code/No-code interface.

outlined in deliverable 2.6. The system is composed of three logical parts, which correspond to the following subsections:

- **deal.II-X Client** (Local client app) - the frontend and main user interface, used to create the computation graph and transparently submit calculations;
- **deal.II-X Cloud** (Remote server) - the web server, managing user accounts and communication with the client;
- **CORAL** - the transducer of the computational graph into executable code.

The complete architecture is reported in Figure 9.

The client acts as a starting point for every new project. Every new computational graph is created or loaded into the client choosing from a list of available computational nodes. The client will upload each computational graph to the HPC cluster and send the command to start the computation. The state of the computation will be displayed in the client. The client will also save the graph to the remote server and the work will be sharable with other users. The HPC cluster and the cloud will never communicate directly but always throughout the client.

2.6.2 deal.II-X Client

The client application provides a no-code graphical interface able to build an editable graph which will then be used to run the simulation in the computing cluster. The graph will be saved and shared among users throughout the remote server. Right

now the application is able to import a set of different available nodes and allow the user to draw with them a MWE (Minimal Working Example) that can be exported via SSH secure connection for execution. It can also send simple commands via SSH and send simple HTTP requests. Those features are needed to connect with the computing cluster and with the remote server where Coral will live. The client app is a native cross-platform application running on the user PC. This approach has been chosen over a fully cloud based application, because it provides few key advantages.

- Streamlined configuration of the connection to the computing cluster. In particular a direct VPN connection can be established leveraging the connection the user normally uses. SSH configuration is also simplified, with no need for additional tunneling from/to the remote server.
- Enhanced security. Users are clearly identified and sensitive data is locally handled, which may be also a requirement for research institutions and other actors.

Technologies. Electron is used to provide native and cross-platform capabilities to the client app. It provides native functionalities from the operating system together with a chromium based environment where any web technology can be used. In particular Svelte and Svelte Flow have been chosen to build the graphical interface, including the canvas from which the graph with nodes and edges is displayed and edited.

Right now a first basic version can be distributed for MacOS and Linux systems or tested in development mode. Core functionalities already available are:

- Importing of different basic set of nodes (from a JSON file adhering to a specific protocol)
- Exporting the graph via SSH (as a JSON file)
- Basic validation on input value and on new edge connection
- Basic HTTP requests towards the remote server
- Deletion of specific nodes

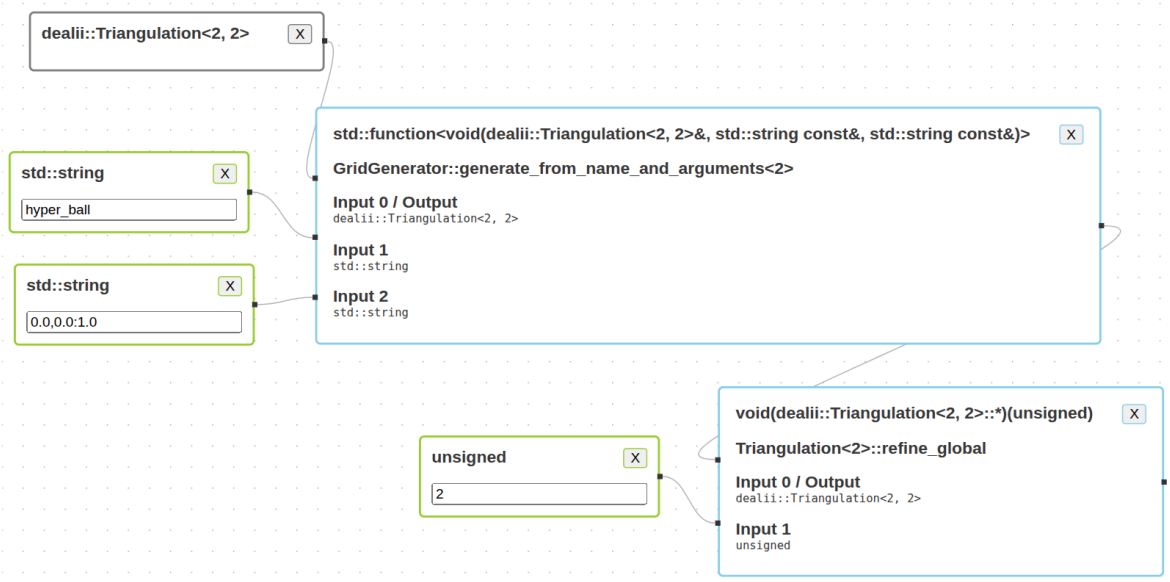

Figure 10: A Minimal Working Example (MWE) graph in the client app.

- Drag&drop of new nodes
- Light and dark modes

2.6.3 deal.II-X Cloud

The deal.II-X Cloud server will be the core of the collaborative nature of the deal.II-X platform, designed to allow sharing of node-based projects between users. As such, it will sport user authentication, project creation and project sharing. The activities on this component will start in the following months.

2.6.4 CORAL

CORAL is a C++ library for building, connecting, and executing computational graphs. It provides a flexible framework for representing computational workflows as directed graphs where nodes represent data or operations, and edges represent dependencies. The library is designed with parallel scientific computing in mind.

In CORAL, every node is designed to be interpreted as a function, represented by its `operator()`. This functional philosophy ensures that nodes are not merely containers of data or operations but are active participants in the computational

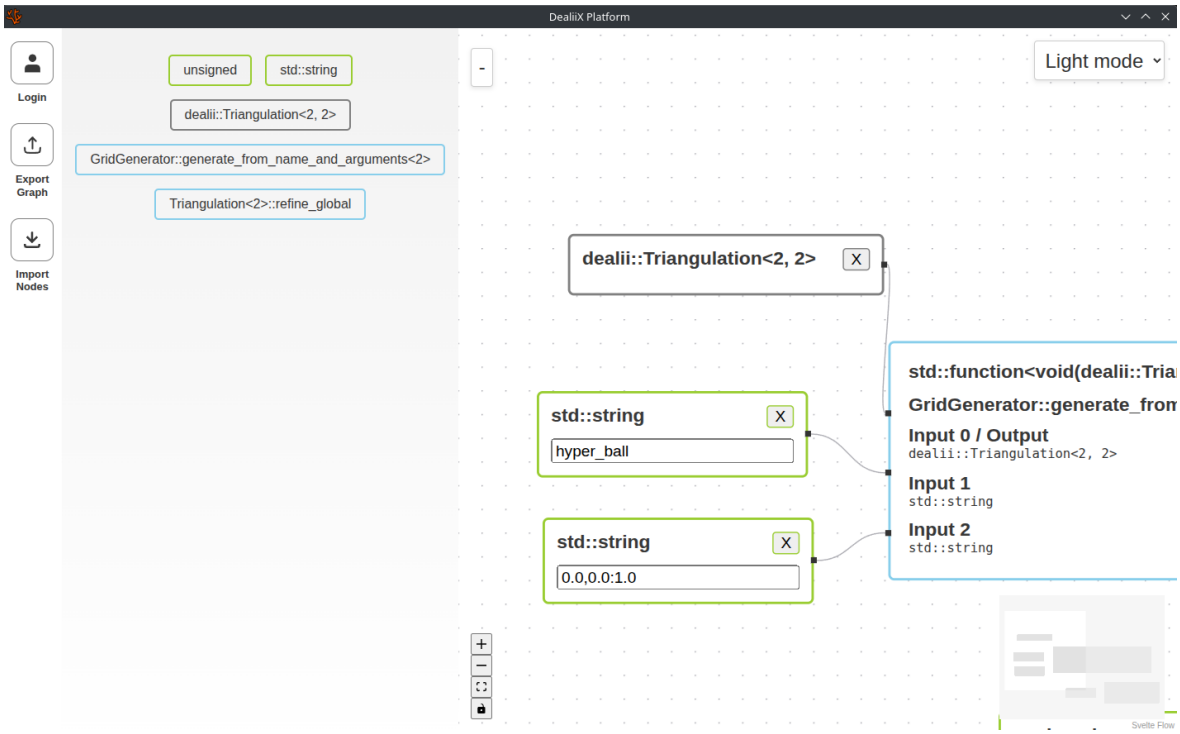

Figure 11: A preview of full interface of the client app.

graph. The execution of a node's function is determined by its type, which defines its behavior and role within the graph.

In the software stack of the deal.II-X, Coral acts as the transducer of the computational graph generated by the Client application into executable instructions. As such, it runs on the computational infrastructure selected by the user (e.g., an HPC cluster).

The main features of CORAL are:

- **Node as a Function**: each node encapsulates a callable function through its operator(). When invoked, the node performs its designated operation, which may include constructing an object, modifying inputs, or producing outputs.
- **Type-Driven Behavior**: the behavior of a node during execution is controlled by its type. For example:
  - **Constructor Nodes**: these nodes create new objects when executed.
  - **Pass-Through Nodes**: these nodes modify their inputs and pass them to their outputs.

  - **Method Nodes**: these nodes invoke a specific method on an object, potentially modifying its state or producing a result.
  - **Function Nodes**: these nodes execute a free function, using their inputs as arguments and producing outputs.

- **Lazy Evaluation**: nodes are executed only when their outputs are explicitly required by other nodes or the user. This ensures that computations are performed efficiently and only when necessary.
- **Input and Output Management**: each node manages its inputs and outputs through a type-safe system. Inputs are connected to other nodes' outputs, and the execution of the node ensures that the outputs are updated accordingly.

This functional approach ensures that nodes in CORAL are versatile and adaptable, capable of representing a wide range of computational tasks. By interpreting nodes as functions, CORAL provides a consistent and intuitive framework for building and executing complex computational workflows.

The core design principles of CORAL are:

- Type Safety
- Reflection System
- Polymorphism Support
- Lazy Evaluation
- Serialization to JSON

The execution model is based on the Taskflow (`https://taskflow.github.io/`) library:

1. Nodes are registered as tasks in a taskflow graph
2. Dependencies between tasks are established based on node connections
3. When execution is requested, tasks are run in the correct order (potentially in parallel)
4. Results flow through the network as tasks complete

The approach based on graph representation embedded into taskflow allows automatic parallelization of tasks when the graph logically allows it.

3 Conclusion

In this document, we have reported the progress made in the digital twin applications of the deal.II-X project.

References

Pasquale Claudio Africa, Roberto Piersanti, Francesco Regazzoni, Michele Bucelli, Matteo Salvador, Marco Fedele, Stefano Pagani, Luca Dede', and Alfio Quarteroni. lifex-ep: a robust and efficient software for cardiac electrophysiology simulations. *BMC Bioinformatics*, 24(1):389, October 2023. doi: 10.1186/s12859-023-05513-8.

Pasquale Claudio Africa, Ivan Fumagalli, Michele Bucelli, Alberto Zingaro, Marco Fedele, Luca Dede', and Alfio Quarteroni. lifex-cfd: An open-source computational fluid dynamics solver for cardiovascular applications. *Computer Physics Communications*, 296:109039, March 2024. doi: 10.1016/j.cpc.2023.109039.

Marco Fedele, Roberto Piersanti, Francesco Regazzoni, Matteo Salvador, Pasquale Claudio Africa, Michele Bucelli, Alberto Zingaro, Luca Dede', and Alfio Quarteroni. A comprehensive and biophysically detailed computational model of the whole human heart electromechanics. *Computer Methods in Applied Mechanics and Engineering*, 410:115983, May 2023. doi: 10.1016/j.cma.2023.115983.

Luca Heltai and Alfonso Caiazzo. Multiscale modeling of vascularized tissues via nonmatching immersed methods. *International Journal for Numerical Methods in Biomedical Engineering*, 35(12):e3264, 2019.

Luca Heltai and Paolo Zunino. Reduced lagrange multiplier approach for non-matching coupling of mixed-dimensional domains. *Mathematical Models and Methods in Applied Sciences*, 33(12):2425–2462, 2023.

Luca Heltai, Alfonso Caiazzo, and Lucas O Müller. Multiscale coupling of one-dimensional vascular models and elastic tissues. *Annals of biomedical engineering*, 49(12):3243–3254, 2021.

Steven A. Niederer, Eric Kerfoot, Alan P. Benson, Miguel O. Bernabeu, Olivier Bernus, Chris Bradley, Elizabeth M. Cherry, Richard Clayton, Flavio H. Fenton,

Alan Garny, Elvio Heidenreich, Sander Land, Mary Maleckar, Pras Pathmanathan, Gernot Plank, José F. Rodríguez, Ishani Roy, Frank B. Sachse, Gunnar Seemann, Ola Skavhaug, and Nic P. Smith. Verification of cardiac tissue electrophysiology simulators using an *N* -version benchmark. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 369(1954): 4331–4351, November 2011. doi: 10.1098/rsta.2011.0139.

Roberto Piersanti, Pasquale C. Africa, Marco Fedele, Christian Vergara, Luca Dedè, Antonio F. Corno, and Alfio Quarteroni. Modeling cardiac muscle fibers in ventricular and atrial electrophysiology simulations. *Computer Methods in Applied Mechanics and Engineering*, 373:113468, January 2021. doi: 10.1016/j.cma.2020.113468.

Alberto Salvadori, Mattia Serpelloni, and Robert M McMeeking. Chemo-thermo-mechanics for mixtures of solids generated by a chemical reaction within a liquid. *to be submitted to: Journal of the Mechanics and Physics of Solids*, 2025.

Richard Schussnig, Niklas Fehn, Peter Munch, and Martin Kronbichler. Matrix-free higher-order finite element methods for hyperelasticity. *Computer Methods in Applied Mechanics and Engineering*, 435:117600, February 2025. doi: 10.1016/j.cma.2024.117600.

Alberto Zingaro, Christian Vergara, Luca Dede', Francesco Regazzoni, and Alfio Quarteroni. A comprehensive mathematical model for cardiac perfusion. *Scientific Reports*, 13(1):14220, August 2023. doi: 10.1038/s41598-023-41312-0.