ZURICH, SWITZERLAND
6 - 10 OCTOBER 2025

EUROPEAN GEOTHERMAL CONGRESS 2025

WBGeo - Workbench for Digital Geosystems

Jan von Harten[1], Marzieh Baes[2], Alexander Lüpges[3], Florian Wellmann[1,2], Mauro Cacace[2], Denise Degen[2,4], Magdalena Scheck-Wenderoth[2], Bernhard Rumpe[3] and Jan Niederau[5]

[1]Chair of Computational Geoscience, Geothermics and Reservoir Geophysics, RWTH Aachen University, Mathieustr. 30, 50274 Aachen, Germany

[2]GFZ Helmholtz Centre for Geosciences, Telegrafenberg, 14473 Potsdam, Germany

[3]Chair of Software Engineering, RWTH Aachen University, Ahornstraße 55, 50274 Aachen, Germany

[4]Institute of Applied Geosciences, TU Darmstadt, Schnittspahnstraße 9, 64287 Darmstadt, Germany

[5]Fraunhofer IEG, Fraunhofer Research Institution for Energy Infrastructures and Geotechnologies IEG, Aureliusstraße 2, 52062 Aachen, Germany

jan.vonharten@cg3.rwth-aachen.de

Keywords: Structural geological modeling, Meshing, Process simulation, Software architecture.

ABSTRACT

In geothermal energy development, decision-making relies on complex analyses to evaluate the feasibility, efficiency, and sustainability of potential projects. These analyses integrate geological, technological, economic, and environmental factors to ensure data informed outcomes.

A key element is the use of structural geological models, which provide detailed representations of subsurface formations. These models subsequently support reservoir process simulations that are essential for predicting the performance of geothermal systems. However, the workflows to create structural models and perform process simulations are often tailored to specific scenarios, requiring extensive expert knowledge, multiple software tools, technical skill and frequent manual adjustments. While this is a consequence of the inherent complexity of geological formations and the variability of subsurface processes, it often limits reusability, comparability, reliability, and reproducibility of such workflows.

To overcome these limitations, we develop a workbench for digital geosystems. This workbench integrates three core components of the described workflows: structural geological modeling, numerical process simulation, and visualization, along with all connecting interfaces. A visual scripting environment built on a domain-specific language provides intuitive access for users with limited technical expertise, while a flexible, modular structure ensures experienced users can access and modify the underlying code. Components of the workbench include a variety of different solutions for each part of a workflow that are completely interchangeable.

We evaluate our workbench using a simple structural model, exploring different interpolation methods and generating watertight structured and unstructured meshes. A thermal conduction model is developed and its results are visualized to test the full workflow.

1. INTRODUCTION

Creating geoscientific models is a complex process. To adequately address specific scenarios, modelers often need to develop workflows that encompass a diverse range of scientific fields. This requires the collaboration of experts from various domains, as well as the integration of multiple software solutions and the necessary skill sets to construct intricate workflows. These workflows typically involve numerous model-specific choices, manual adaptations, and software-specific formats. While this complexity is inherent in geoscientific modeling, it also leads to several challenges in the modeling process (Wellmann & Caumon, 2018):

1. Reproducibility: The amount of manual adjustments required throughout these workflows can significantly hinder reproducibility across different studies.

2. Comparability: Due to time and financial constraints, it is often challenging to compare multiple methods for a single model, limiting the evaluation of alternative approaches.

3. Reusability: Workflows designed for specific scenarios that include manual adjustments are challenging to reuse for new situations; often only portions of the workflow can be repurposed.

4. Software: Users may be accustomed to certain software tools due to either their skill level or the access provided by their institution, which can restrict exploration of alternative methodologies.

5. Programming Skills: Depending on the complexity of the workflow and the software solutions employed, basic or even advanced programming skills may be required.

In this study we address the aforementioned challenges by developing a workbench - rather than a singular software solution - that integrates a set of interchangeable components for each step of the workflow. This setup is grounded in a Component and Connector (C&C) architecture along with domain-specific languages (DSLs) (Ringert et al., 2014).

Our workbench is based on a typical geoscientific modeling workflow that begins with the creation of a structural geological model based on a set of input data. The resulting model is then translated into watertight volumetric meshes, which provide the computational basis for subsequent process simulations

The resulting workbench establishes fixed interface formats, meaning a predefined structure for data exchange between software components, ensuring compatibility and consistency in communication, for each stage of the workflow. A variety of components perform essential modeling steps; multiple options are available for each step and can be utilized interchangeably. This flexibility allows users to easily compare different methods and develop workflows based on various approaches to evaluate results without requiring extensive knowledge of all methodologies or advanced programming skills. It further improves reusability and reproducibility of created models, as necessary tools are available from within the workbench.

In the following sections, we present an initial prototype of the workbench, describe the workflow components and their defined interface formats, and showcase results from a simplified synthetic model.

2. STATE OF THE ART

Domain-specific languages (DSLs) have been successfully applied in various fields. Examples in academic settings are the application of DSLs for specifying Internet of Things (IoT) systems (Kirchhof et al., 2022) and in robotics (Adam et al., 2017). In industry, DSLs are employed in software development, including cloud resource management and hardware design (VHDL). DSLs are typically created using language workbenches (Erdweg et al., 2013) such as JetBrains MPS (Pech et al., 2013), Eclipse Xtext (Efftinge & Völter, 2006), or MontiCore (Hölldobler et al., 2021). These workbenches facilitate the creation of tools for reading and processing models based on language definitions.

In geosciences, DSLs have primarily been utilized in high-performance computing (HPC) for rapid adaptation to new hardware infrastructures. Generic DSLs like Kokkos (Carter Edwards et al., 2014) and RAJA (Beckingsale et al., 2019), as well as geoscientific DSLs such as GridTools (Afanasyev et al., 2021) and CLAW DSL (Clement et al., 2018), are examples.

Component & Connector (C&C) architectures serve as an abstraction for modeling the runtime structure of software systems (Medvidovic & Taylor, 2000). They are commonly applied in domains such as microservices (Söylemezet al., 2024) and cyber-physical systems (Lee, 2008). The structure of software systems based on C&C architectures is often specified using a specialized form of DSL, known as an architecture description language (Vestal, 1993).

In recent decades, various algorithms for geological modeling have been developed (Wellmann & Caumon, 2018). Recent progress in implicit methods has allowed a high degree of automation in the modeling of complex 3D structures. On one hand, there are commercial tools that often include internal extensions for geothermal process simulations and may feature Application Programming Interfaces (APIs) to facilitate limited integration as components within workflows. On the other hand, open-source alternatives such as GemPy (de la Varga et al., 2019) and LoopStructural (Grose et al., 2021) offer similar modeling approaches while providing full access to all functionalities, making them suitable to be seamlessly integrated into automated workflows.

A variety of algorithms and software tools have been developed to generate high-quality meshes from structural geological models characterized by irregular and heterogeneous structures. Widely used meshing tools in geosciences include Gmsh, TetGen, and CUBIT (Blacker et al., 1994; Geuzaine & Remacle, 2009; Si, 2007). Despite recent progresses, automated mesh generation from geological models remains challenging. Traditional methods for meshing complex geological structures require substantial manual effort, being only partially automatable. Currently, no fully automated method exists to generate meshes based on input from implicit geological models. One of the major obstacles in generating watertight meshes stems from gaps and overlapping surfaces, as usually introduced by Marching Cubes algorithm. As a result, integrating geological modeling software with simulation environments is typically done manually or through custom-written wrappers (Blessent et al., 2009; Niederau et al., 2022; Wellmann et al., 2012).

3. METHODS

3.1 Combined workflow and interface formats

The primary challenge in establishing the workbench stems from the definition of interface formats. We delineate these formats by specifying some mandatory information alongside optional information. Data is organized within object classes in various formats. Mandatory information is essential for each component available for a specific modeling step, while optional information may either be required or utilized by certain components. The provided framework can easily be expanded and adapted in the future to integrate new components and evolving requirements.

This adaptability not only supports current needs but also promotes long-term sustainability by allowing for future enhancements and modifications.

It is important to note that not all components require the same data, nor can all components accommodate every scenario. The workbench is designed to provide feedback that informs users about which components are compatible based on the provided input data.

3.2 Software architecture

Workflows are designed using a C&C architecture (Medvidovic & Taylor, 2000). In this framework, components perform computations, with their interfaces consisting of input ports and output ports (I/O ports). Data is exchanged through connectors that link these ports. C&C architectures effectively describe the overarching structure of software systems and enable the construction of modular, reusable workflows by combining various components.

Since only the interface of a component is required for modeling workflows, the underlying implementation remains independent of the workflow itself. While it is feasible to substitute one component implementation for another, different components in this setting require distinct tuning input parameters; consequently, the interfaces for similar components in the workbench may differ.

The ports—and thus the connectors—are typed to primitive data types, such as numbers or the interface formats described in Section 3.1. This type safety ensures that modelers can design correct workflows. The ability to define workflows is further enhanced by a DSL. The architecture of this project provides both a textual (H¨olldobler et al., 2021) and a visual representation (Harel, 1988). DSLs allow workflows to be defined, archived, and discussed independently of a general purpose language and component implementation.

In contrast to traditional C&C visualizations, the connectors in this project display their values as object components. This feature eases the export and inspection—such as visualization—of their values. By employing a C&C architecture, computation across components can be executed within a distributed infrastructure, facilitating pre-computation and caching capabilities.

3.1 Structural geological modeling components

The first interface format defined in the workbench pertains to the input data required for creating a structural geological model. The mandatory information includes the name of the model, its extent, and a resolution for a regular grid. Additionally, surface points that delineate the bottom surfaces of structural elements (e.g., rock units) are required, along with an ordering and grouping of these elements. Optional information encompasses the orientation of surfaces and details regarding faults and fault relationships.

Although the interpolation methods for these components may vary, the general structure remains consistent: Elements that are grouped together represent concordant units and are interpolated using a single implicit scalar field, resulting in semiparallel behavior. The order of groups defines their interactions; younger groups erode older ones, thereby creating unconformities.

The current iteration of the workbench includes four distinct components for interpolating structural geological models. All these components feature implicit interpolators and implementations based on available open-source software.

We have incorporated two basic interpolators that are not specifically tailored to structural geological modeling: Radial Basis Function (RBF) implemented in SciPY (Virtanen et al., 2020) and Ordinary Kriging (OK) implemented in PyKrige (Murphy et al., 2024). At this stage, these interpolators cannot accommodate models that include faults and do not account for orientation data. The other two available components for this step are Universal Co-Kriging (Calcagno et al., 2008; Lajaunie et al., 1997) and an implicit neural network approach. Universal Cokriging is a kriging method that integrates orientation data and allows for fault inclusion by applying a drift function. The implementation featured in the workbench is based on GemPy (de la Varga et al., 2019). Additionally, we include an approach to creating structural models using implicit neural networks implemented in GeoINR (Hillier et al., 2023), which does integrate orientation data.

It is important to note that all these methods involve a set of additional parameters—such as variogram specifications—that can be optimized for specific scenarios and passed when selecting a component. We also want to emphasize that interpolators with different requirements and capabilities were chosen on purpose to showcase and take advantage of the modularity offered by the framework.

The output of the structural geological model—and thus the second interface format—includes a lithology block, which assigns a rock ID to each location within the regular grid, as well as surface meshes that define spatial boundaries between elements. Optionally, the resulting underlying scalar fields and gradients can be stored for use in subsequent workflow steps.

3.1 Meshing components

So far, we have developed two components for generating explicit structured and unstructured meshes based on the results of the structural geological model. A structured mesh is created by first generating a regular rectangular grid from the coordinates of the surface boundary meshes from the previous workflow step. We use RBF interpolation to generate grid points which are uniformly distributed across all layers to ensure consistent spacing throughout the model. Next, hexahedral grids are created by connecting grid points

between layers and physical properties of each layer are assigned by associating a unique layer id to each element. In our structured meshing method, the number of grids per layer is user-defined. Since this method maintains a constant grid count per layer across the entire domain, it may result in very small/overlapping grids in areas where layers are closely spaced. To prevent this, we set a minimum grid size threshold in the z-direction – which can be optimized by the user - and iteratively adjust grid points upward when necessary. Since modifying one grid point affects neighboring grid sizes, this process is repeated until all grids meet the threshold. Due to its assumption of uniform grid points across layers, this method is most suitable for models with predominantly horizontal and not overlapping geological layers.

The second meshing component included in the workbench is an unstructured meshing approach. Here, grids are created from point clouds obtained from the structural geological model, without requiring uniform grid points across layers. These grids are imported into Gmsh, which is an open-source three-dimensional finite element mesh generator (Geuzaine & Remacle, 2009), where B-spline surfaces are generated. We then compute all intersections among the input surfaces to constrain the final model volume. Physical properties are assigned to each sub-volume according to its corresponding geological layer, and tetrahedral elements are generated with user-defined mesh sizes. While this method allows for more complex geological structures, it requires rectangular grids to construct B-spline surfaces, making it less suitable for models with convoluted geological manifolds.

The resulting meshes are stored as elements and nodes and optionally in a Pyvista Multiblock for visualization (Sullivan & Kaszynski, 2019). The interface also supports exporting the generated meshes in multiple formats, including Exodus, VTK, and VTM, ensuring compatibility with various simulation tools.

3.1 Process simulation components

The process simulation component targeted in this study is provided by the multiphysics open source simulator GOLEM (Cacace & Jacquey, 2017). GOLEM is an object-oriented, scalable, and implicit finite element simulator based on the MOOSE framework (Giudicelli et al., 2024) tailored to simulate thermo-hydromechanical-chemical (THMC) processes in fractured and porous rocks. A feature of interest for the targeted applications is its geometric agnosticism, which enables to model faults/fractures and geothermal wells as lower dimensional discontinuous frictional elements embedded in a 3D porous reservoir. Interfacing GOLEM to the meshing component relies on the export functionalities of the final mesh into an exodus (netcdf) format.

4. RESULTS

. We test our workbench by performing the steps of the presented workflow on a synthetic model with basic geological features that are characteristic for many geological modeling applications. The chosen workflow created with the graphical view of the DSL is shown in Figure 1.

The model covers an area from 0 to 2000 arbitrary units in the x-direction, from 0 to 1000 units in the y-direction, and from 0 to 1000 units in the z-direction. Although it is technically three-dimensional, its consistent behavior in the y-direction leads to a simplified 2.5D behavior.

The input data object includes information about the locations of surface points for several structural elements as shown in Figure 2A. We specify a desired resolution for a regular grid used for discretization of

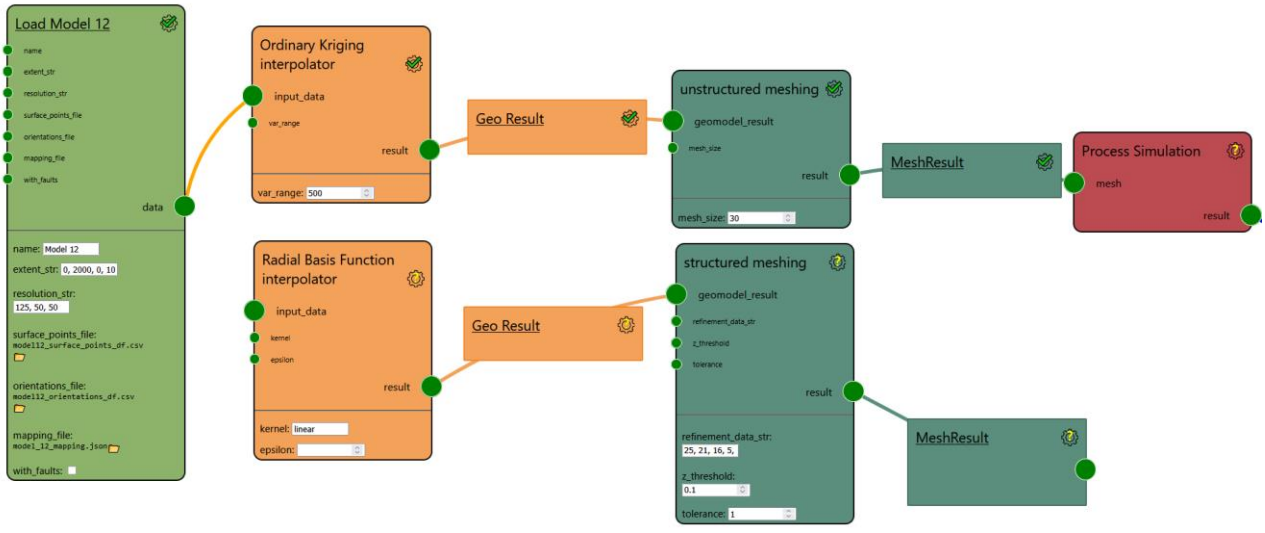

Figure 1: Screenshot of a workflow created in the prototype of the graphical DSL. Components are color coded based on their step in the workflow: Light green: Interface object for input data, Orange: Structural geological modeling, Dark green: Meshing, Red: Process simulation). Note how the workflow at the top can seamlessly be replaced by the bottom workflow by moving the connections.

the structural geological model space. Furthermore, we define the relationships among existing structural elements—from younger to older and group them into structural groups to ensure semi-parallel alignment using a mapping object. It is important to note that this mapping is based on our subjective interpretation of the input data. For this model we group the two deep elements rock 1 and rock 2 together, while the overlying elements rock 4 and rock 3 form a group that erodes the underlying elements.

In the next step we create a structural geological model based on the input data using one of our available modeling algorithms. In this example the Ordinary Kriging interpolator is used. Note, however, that because the model does not include faults and orientations are optional, all four existing interpolation components for structural modeling are viable and produce similar results, as this synthetic model is well-constrained. The resulting boundaries can be seen in Figure 2B.

Next, we generate both structured and unstructured meshes based on results of the structural geological model (Unstructured mesh shown in Figure 2C). To fully evaluate performance of our workbench, we then develop simple thermal conduction simulations for both mesh types. The structural model consists of five distinct layers, each with different thermal conductivities. The thermal boundary conditions are set to Dirichlet boundary conditions such that the temperature at the top boundary is 10 °C, and the temperature at the base boundary is 40 °C.

The initial temperature field within the model following a standard geothermal gradient is governed by the equation:

$$T(z) = -0.03(z - 1000) + Tp \qquad [1]$$

where Tp is the temperature at the top boundary and z is the depth. The resulting temperature field for simulation with the unstructured mesh is shown in Figure 2D). Note that simulation results for both mesh types reveal that the resulting temperature profiles are nearly identical, indicating the robustness and convergence of our meshing methods.

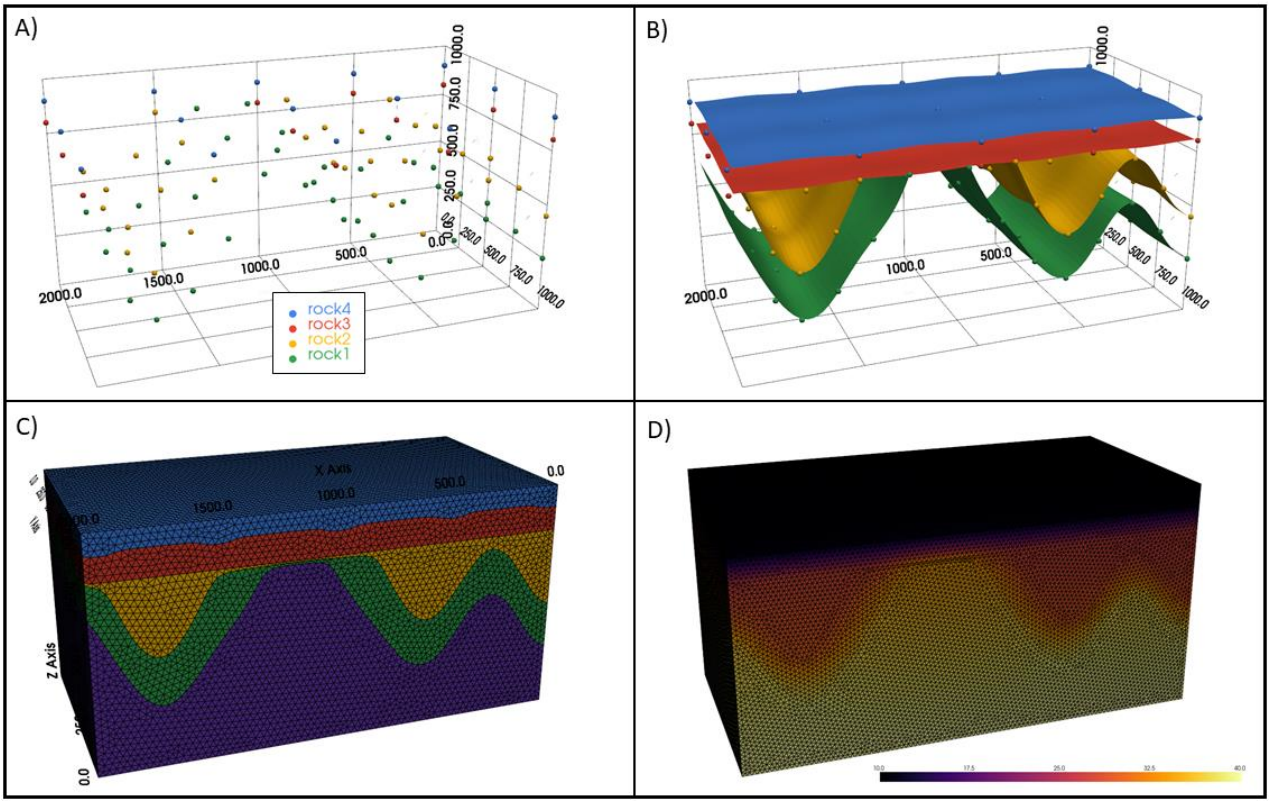

Figure 2: Results for the full workflow presented in Figure 1. A) Input data: Surface points for multiple rock elements. B) Structural geological model, represented by surface meshes, created using Ordinary Kriging component. C) Unstructured volumetric mesh automatically created from the structural model. D) Temperature field results from a simple thermal conduction simulation.

5. DISCUSSION

The framework and workbench developed within this project presents an innovative approach to streamline geoscientific workflows, addressing several inherent challenges faced by modelers when creating workflows for complex geoscientifc scenarios. While numerous solutions exist for specific steps within these workflows, they often present significant obstacles due to their disparate interfaces. This variability

complicates comparisons between tools and hinders benchmarking efforts. Consequently, practitioners frequently resort to using familiar solutions available at their institutions, which can lead to inconsistencies, limitations in the overall workflow efficiency and evolution of institutional ”islands”.

Many existing solutions also necessitate advanced programming skills, particularly when users seek to integrate multiple tools into a comprehensive workflow. This requirement can deter domain experts who possess limited technical expertise but have valuable insights into geological processes.

Moreover, once established, workflows tend to be highly specific to individual scenarios, rendering them difficult to reuse across different projects or studies. This lack of reusability underscores the need for a more flexible and modular approach that can accommodate varying geological contexts.

It is important to note that the workbench presented here is currently a prototype with several limitations. At this stage, it can only handle simplified structural models; for instance, not all interpolators are capable of managing faults, and many require parameter optimization tailored to specific models (e.g., variogram adjustments for Ordinary Kriging). Additionally, automated meshing capabilities are restricted solely to these simplified models at present. The generation of watertight meshes from more complex structural models - necessary for effective process simulation - is a well-known bottleneck. By leveraging gradient information from scalar fields, we aim to enhance this aspect of workflow development. The integration of proprietary software solutions also poses challenges that must be addressed as we advance the development of this framework.

Despite these constraints, we believe that our workbench serves as an essential prototype for creating a unifying framework aimed at enhancing accessibility of geoscientific tools. By moving away from isolated solutions developed by specific institutions—often used solely within those settings—we aspire to facilitate easier comparisons among methodologies and promote broader collaboration within the research community.

We recognize that achieving these ambitious goals is no small task; however, we are committed to laying the groundwork for a system where new components can be seamlessly integrated as long as they adhere to defined interface formats. Furthermore, by offering both textual and graphical DSLs, we aim to increase accessibility for domain experts who may have little or no programming experience.

5. CONCLUSION

In this work, we have presented the development of a workbench for digital geosystems that addresses the complexities inherent in geoscientific modeling workflows. By employing a C&C architecture with DSLs and defining clear interface formats, our approach enhances modularity, reusability, and accessibility. Future developments will focus on expanding component capabilities for more complex models, adding new components for the varying workflow steps and improving the workbench framework.

REFERENCES

Afanasyev, A., Bianco, M., Mosimann, L., Osuna, C., Thaler, F., Vogt, H., Fuhrer, O., VandeVondele, J. and Schulthess, T. C.: GridTools: A framework for portable weather and climate applications, *SoftwareX*, 15, (2021), 100707.

Adam, K., Hölldobler, K., Rumpe, B. and Wortmann, A.: Modeling Robotics Software Architectures with Modular Model Transformations, *Journal of Software Engineering for Robotics (JOSER)*, 8, (2017), 3–16

Beckingsale, D. A., Burmark, J., Hornung, R., Jones, H., Killian, W., Kunen, A. J., Pearce, O., Robinson, P., Ryujin, B. S. and Scogland, T. R. W.: RAJA: Portable Performance for Large-Scale Scientific Applications, Proceedings of the 2019 IEEE/ACM International Workshop on Performance, Portability and Productivity in HPC (P3HPC), (2019), 71–81.

Blacker, T. D., Bohnhoff, W. J. and Edwards, T. L.: *CUBIT mesh generation environment. Volume 1: Users manual*, Sandia National Lab., Albuquerque, NM, (1994).

Blessent, D., Therrien, R. and MacQuarrie, K.: Coupling geological and numerical models to simulate groundwater flow and contaminant transport in fractured media, *Computers & Geosciences*, 35, (2009), 1897–1906.

Cacace, M. and Jacquey, A. B.: Flexible parallel implicit modelling of coupled thermal–hydraulic–mechanical processes in fractured rocks, *Solid Earth*, 8, (2017), 921–941.

Calcagno, P., Chilès, J. P., Courrioux, G. and Guillen, A.: Geological modelling from field data and geological knowledge: Part I. Modelling method coupling 3D potential-field interpolation and geological rules, *Physics of the Earth and Planetary Interiors*, 171, (2008), 147–157.

Carter Edwards, H., Trott, C. R. and Sunderland, D.: Kokkos: Enabling manycore performance portability through polymorphic memory access patterns, *Journal of Parallel and Distributed Computing*, 74, (2014), 3202–3216.

Clement, V., Ferrachat, S., Fuhrer, O., Lapillonne, X., Osuna, C. E., Pincus, R., Rood, J. and Sawyer, W.: The CLAW DSL: Abstractions for performance portable weather and climate models, in *Proceedings of the Platform for Advanced Scientific Computing Conference (PASC '18)*, Association for Computing Machinery, New York, NY, USA, (2018), 1–10.

de la Varga, M., Schaaf, A. and Wellmann, F.: GemPy 1.0: open-source stochastic geological modeling and inversion, Geoscientific Model Development, 12, (2019), 1–32.

Efftinge, S. and Völter, M.: oAW xText: A framework for textual DSLs, in *Workshop on Modeling Symposium at Eclipse Summit*, 32(118), (2006).

Erdweg, S., Van Der Storm, T., Völter, M., Boersma, M., Bosman, R., Cook, W. R., Gerritsen, A., Hulshout, A., Kelly, S., Loh, A., *et al.*: The state of the art in language workbenches: Conclusions from the language

workbench challenge, in *Software Language Engineering: 6th International Conference, SLE 2013, Indianapolis, IN, USA, October 26–28, 2013. Proceedings 6*, pp. 197–217, Springer (2013).

Geuzaine, C. and Remacle, J.-F.: Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities, *International Journal for Numerical Methods in Engineering*, 79(11), (2009), 1309–1331.

Giudicelli, G., et al.: MOOSE 3.0: Enabling massively parallel multiphysics simulations, *SoftwareX*, 26, (2024), 101690.

Grose, L., Ailleres, L., Laurent, G. and Jessell, M.: LoopStructural 1.0: time-aware geological modelling, *Geoscientific Model Development*, 14(6), Göttingen, (2021), 3915–3937.

Harel, D.: On visual formalisms, *Communications of the ACM*, 31(5), New York, NY, USA, (1988), 514-530.

Hillier, M., Wellmann, F., de Kemp, E. A., Brodaric, B., Schetselaar, E. and Bédard, K.: GeoINR 1.0: an implicit neural network approach to three-dimensional geological modelling, *Geoscientific Model Development*, 16(23), (2023), 6987-7012.

Hölldobler, K., Kautz, O. and Rumpe, B.: MontiCore Language Workbench and Library Handbook: Edition 2021, *Shaker Verlag*, Aachen, (2021), *Aachener Informatik-Berichte, Software Engineering, Band 48*

Kirchhof, J. C., Rumpe, B., Schmalzing, D. and Wortmann, A.: MontiThings: Model-Driven Development and Deployment of Reliable IoT Applications, *Journal of Systems and Software*, 183, (2022), 111087.

Lajaunie, C., Courrioux, G. and Manuel, L.: Foliation fields and 3D cartography in geology: Principles of a method based on potential interpolation, *Mathematical Geology*, 29(4), (1997), 571-584.

Lee, E. A.: Cyber physical systems: Design challenges, in: *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, IEEE, (2008), 363-369.

Medvidovic, N. and Taylor, R. N.: A Classification and Comparison Framework for Software Architecture Description Languages, *IEEE Transactions on Software Engineering*, 26(1), (2000), 70-93.

Murphy, B., Yurchak, R. and Müller, S.: GeoStat-Framework/PyKrige: v1.7.2, *Zenodo*, (2024).

Niederau, J., Samrock, F., Saar, M. O. and Ebigbo, A.: Pilot study to determine the geothermal heat flux distribution in the Canton of Aargau: Final report, *ETH Zurich*, (2022).

Pech, V., Shatalin, A. and Voelter, M.: JetBrains MPS as a tool for extending Java, in: *Proceedings of the 2013 International Conference on Principles and Practices of Programming on the Java Platform: Virtual Machines, Languages, and Tools*, (2013), 165-168.

Ringert, J. O., Rumpe, B. and Wortmann, A.: Architecture and Behavior Modeling of Cyber-Physical Systems with MontiArcAutomaton, *Shaker Verlag*, Aachen, (2014), *Aachener Informatik-Berichte, Software Engineering, Band 20*.

Si, H.: A Quality Tetrahedral Mesh Generator and Three-Dimensional Delaunay Triangulator, *Web Intelligence and Agent Systems: An International Journal - WIAS*, 75, (2007).

Söylemez, M., Tekinerdogan, B. and Tarhan, A. K.: Microservice reference architecture design: A multi-case study, *Software: Practice and Experience*, 54(1), (2024), 58-84.

Sullivan, C. B. and Kaszynski, A.: PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK), *Journal of Open Source Software*, 4(37), (2019), 1450.

Vestal, S.: A cursory overview and comparison of four architecture description languages, *Technical Report, Honeywell Technology Center*, (1993).

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., et al. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*(3), 261–272.

Wellmann, F. and Caumon, G.: 3-D Structural Geological Models: Concepts, Methods, and Uncertainties, in: *Advances in Geophysics*, Schmelzbach, C. (Ed.), 1-121, *Elsevier*, (2018).

Wellmann, F., Croucher, A. and Regenauer-Lieb, K.: Python Scripting Libraries for Subsurface Fluid and Heat Flow Simulations with TOUGH2 and SHEMAT, *Computers & Geosciences*, 43, (2012).

Acknowledgements

This work was supported by the German Federal Ministry of Education and Research under the Geoforschung für Nachhaltigkeit (GEO:N) initiative (Grant number: 03G0922A).