X International Conference on Computational Methods for Coupled Problems in Science and Engineering
COUPLED PROBLEMS 2023
M. Papadrakakis, B. Schrefler and E. Oñate (Eds)

INTRODUCING A CLOUD-BASED FRAMEWORK FOR CREATING, VISUALISING, TESTING AND AUTOMATING COMPLEX SIMULATION WORKFLOWS

Olivia A. Stodieck

Dapta Ltd
Newminster House, 27-29 Baldwin Street, Bristol BS1 1LT, United Kingdom
e-mail: olivia.stodieck@dapta.com, web page: https://www.dapta.com

Key words: Coupled Problems, Multiphysics Problems, Automation, Multidisciplinary Design Optimisation, Open Source Software

Abstract. Engineers increasingly need tools that help them automate complex simulation workflows. Besides performance, robustness and usability requirements, tools should also be easily accessible. To fulfil these requirements, Dapta Ltd is developing a cloud-based framework, which is designed to be an all-in-one solution to create, visualise, test and automate simulation workflows. Here we demonstrate the use of the dapta platform with open-source software libraries, focusing on an FSI multiphysics example.

1 INTRODUCTION

As performance expectations rise, products are becoming more complex and so are engineering simulation workflows, models and processes. In a recent industry report [1], it is estimated that more than half of R&D-driven organisations are using three or more different simulation software packages (not counting in-house tools), in many cases to support multidisciplinary simulation efforts. Since each discipline has a different focus, the modelling approaches and levels of fidelity are rarely consistent, even if the same software package is used. One example of this can be seen in the aerospace industry when considering the effects of aeroelasticity on the structural design of aircraft wing structures. A wing structure is typically idealised using low-fidelity beam elements to compute the aeroelastic load re-distributions, but the actual wing sizing analysis requires the use of more complex finite element models, coupled with the evaluation of numerous strength and design criteria [2, 3].

In the past, the task of transferring and processing data between different loosely coupled simulations has been a largely manual process. However, this approach is incompatible with the increasing complexity of the models, the desire for faster design cycles and more robust performance predictions. Indeed, standardisation and automation of these manual processes is key to the application of formal multidisciplinary design optimisation approaches. To illustrate this concept, figure 1 from reference [3] shows a possible flow of data between a gradient-based design optimisation algorithm and aeroelastic and structural wing model analyses.

The requirement for automating data processing steps is also present in multiphysics simulation problems where different numerical methods are used, such as the Eulerian finite volume

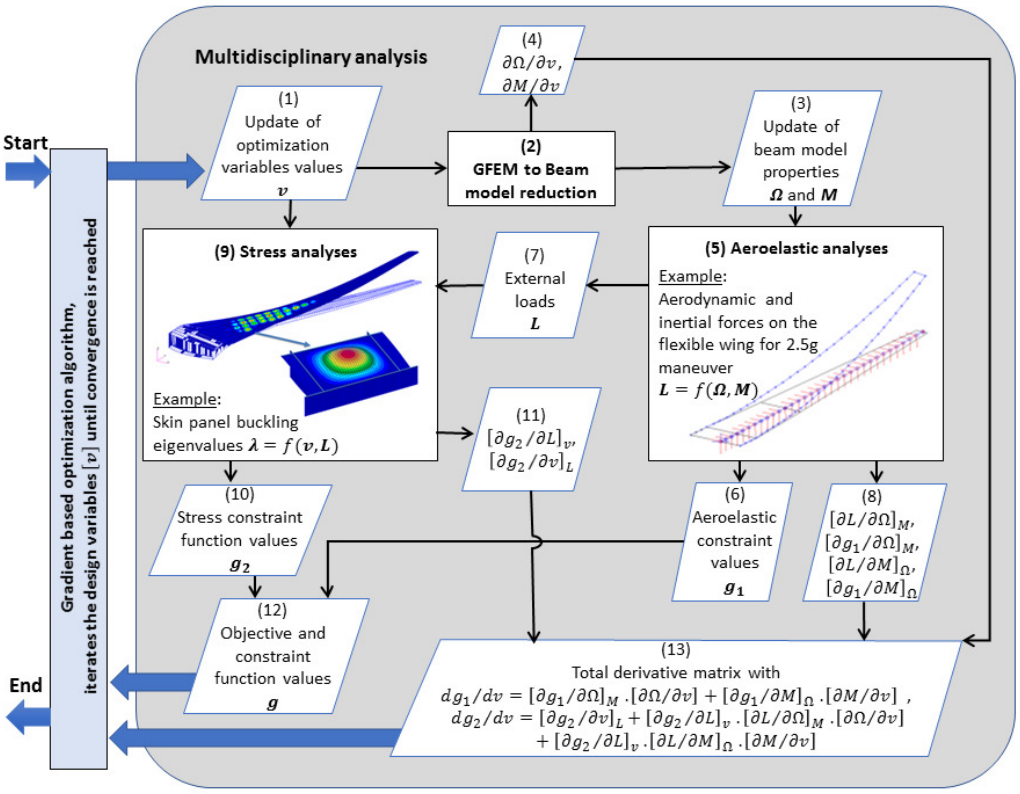

Figure 1: Aeroelastic optimization with detailed stress constraints, reproduced from [3]

method for flow simulation and the Lagrangian finite element methods for the structural dynamics. In this case, the physics coupling is not intrinsic to the problem solution process and instead needs to be implemented through purpose-built solver interfaces and coupling libraries.

The general requirements for multiphysics and loosely coupled simulation processes are deemed to be similar enough here to combine them in the discussion that follows. In the first section, we introduce the new dapta web platform [4], and describe how it addresses the requirements outlined above. The main user tasks that can be performed using the platform are described in the second section. In the last section, we provide an insight into the platform's current use with various open-source software (OSS) tools and present a specific fluid-structure interaction (FSI) multiphysics application example, that uses the preCICE library [5] to couple the finite element solver CalculiX CrunchiX [6] to the computational fluid dynamics solver OpenFOAM [7].

2 A cloud-based simulation workflow automation framework

In January 2020, the new dapta platform prototype was launched [4], allowing users to create and automate complex simulation workflows in the cloud. The platform currently provides easy access to web-based simulation capabilities, using python interfaces and mainly open-source software tools for research purposes. The platform aims to provide:

- Compatibility with a wide range of existing software tools and services;

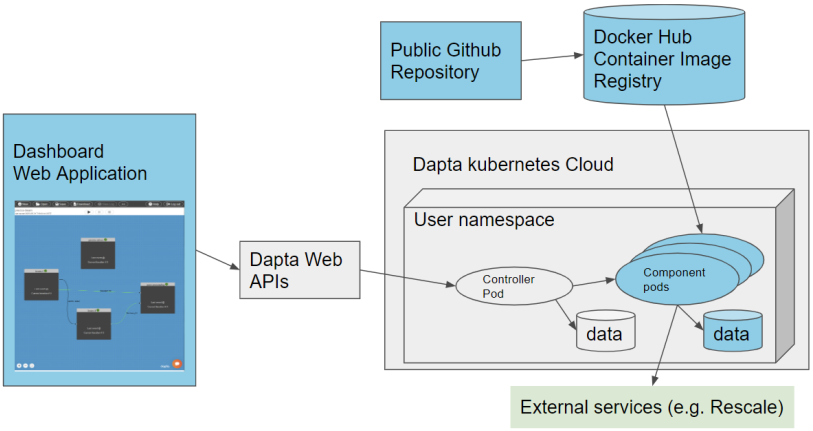

Figure 2: A modular architecture

- Fully customisable and version controlled workflows and simulation interfaces;
- Secure, highly available and scalable cloud resources;
- User-friendly web interfaces for self-service simulation anywhere, anytime;
- Collaboration tools for engineers, managers and suppliers.

The platform is built using a modular architecture typical of modern web services, where the main modules are shown in figure 2. Since the user accesses the platform via a secure web-page, the Dashboard shown on the left, there is no requirement to install any local software packages. Indeed, the user could choose to create, view or execute workflows from a smartphone or tablet device. The Dashboard communicates with a kubernetes cloud environment via web APIs, which allow the user to dynamically set up and control compute resources within his/her own user kubernetes namespace.

The modular architecture not only allows features to be added incrementally and released frequently. It also allows users to only load the tools and interfaces that they specifically require, which speeds-up executions and simplifies the user interfaces. Users are also encouraged to download the generic component image templates from the publicly accessible Dapta GitHub page[1] to develop and test more complex simulation components locally, before integrating them into the cloud environment.

3 How to use the platform

Users can create, visualise, test and automate workflows directly from the web interface. The following subsections describe these tasks in more detail.

3.1 Creating a workflow

Once signed-into the dapta platform, each users can work on multiple workflows simultaneously by opening additional web browser tabs. Each workflow is a combinations of linked

[1] https://github.com/daptablade/generic-python3-comp

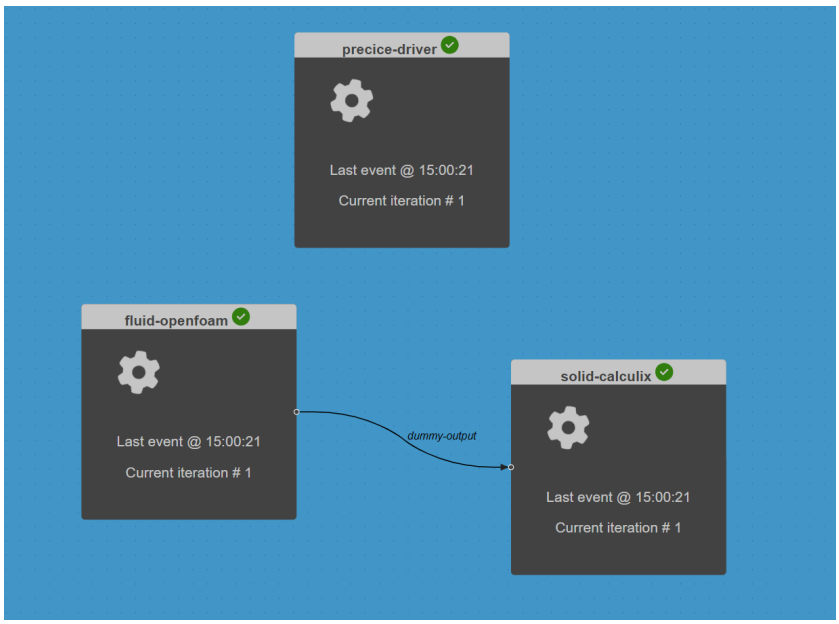

Figure 3: An FSI mutiphysics workflow as it appears in the dapta interface

simulation components, each component being representative of a discrete analysis task, which has its own set of inputs, outputs and parameters. The FSI workflow from section 4 is shown in figure 3. Components are displayed as simple rectangular boxes in the workspace and can be either created directly in the interface by right-clicking or by importing custom JSON input files. Connections between components appear as labelled curved arrows that can be drawn from one component output to another component input. Connections can be used to transfer files or data in the form of numerical arrays, which makes this data easily accessible to workflow drivers such as optimisation algorithms (see 3.4). The user's first task is to define which components he/she wishes to include in the workflow and which software tools or libraries each component should be able to access. The current list of publicly available component images can be found in the user manual[2].

3.2 Visualising workflow inputs

Each component in the workflow requires a minimum number of user inputs before it can be executed. Typically, this information includes: a name, a chosen container image (depending on the software required), a setup script, a computation script and any parameters and custom input files that the setup or computation scripts may need to access. Python templates for the setup and computation scripts are provided in the user manual tutorials. Scripts can be viewed and edited in standard text editors, whilst parameters, inputs and outputs can be accessed and defined directly in the web interface. It is also possible to import or export whole workflows from the dapta web interface in a JSON format, including component parameters, inputs and outputs. A green tick symbol indicates when a component is fully defined and ready for execution.

[2]https://daptadocs.com

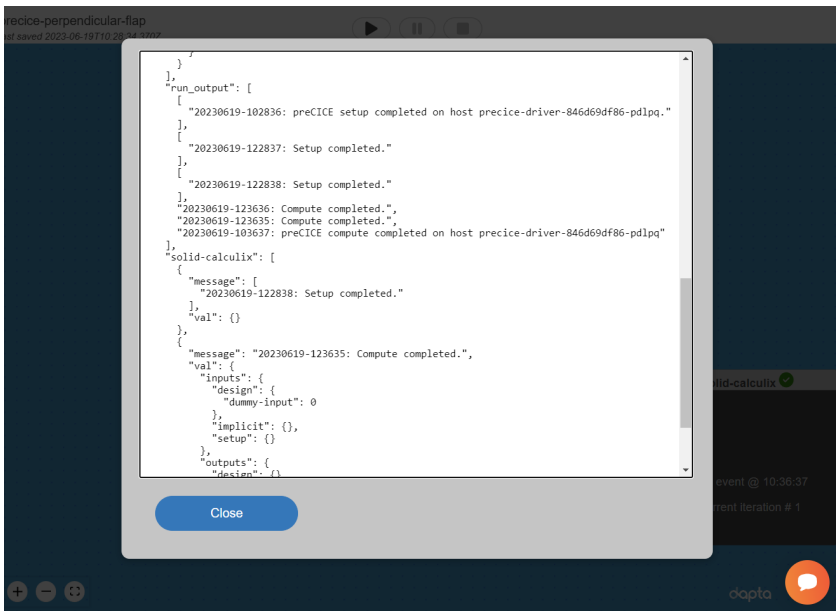

Figure 4: Viewing a run log from the FSI example in section 4

3.3 Testing and viewing outputs

Once all components and connections have been fully defined, the workflow can be executed. Visual clues are used to notify the user of the current status of each component. The cog symbols in figure 3 for example indicate that all three components in the workflow are currently active and in the 'compute' phase. If a component causes an error to be raised, a flashing warning symbol will replace the cog symbol of that component and the workflow execution with stop. The user can view details of the error message in the component's Log tab, where a hyperlink also permits a snapshot of all the latest input and output files generated by the component to be downloaded for further local investigation. Alternatively, the user can view and download the run log at any time after the workflow execution has started. The run log (figure 4) contains a summary of all component log messages (customisable by the user) that are output once the component completes the 'setup' or 'compute' phase. Users are encouraged to define single component workflows initially, to test that these execute as expected before integrating multiple components into more complex workflows.

3.4 Automation

Automation in the dapta platform covers multiple aspects. The first type of automation exists where multiple components are executed in a logical order, in series or in parallel, depending on each component's inputs, outputs and connections to other components. To unambiguously define the order of execution of all components, the user also defines a starting and ending component. Each component waits until its specific required inputs are available before staring a compute phase. The second type of automation is achieved by defining so called workflow 'driver' component. These include design exploration and optimisation drivers, as well as multiphysics simulation drivers that have the ability to launch the execution of specific components or whole workflows once or multiple times. Drivers are also components and they require a similar set of customisable user inputs before they can be executed.

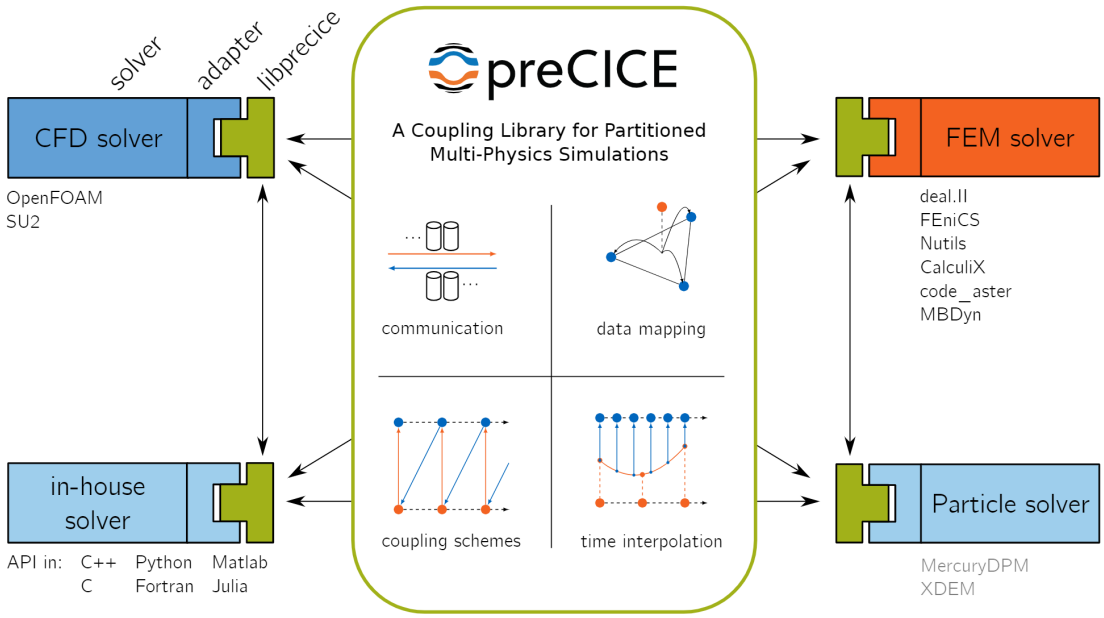

Figure 5: Overview of the preCICE coupling library functionality [5]

4 A multiphysics simulation workflow example

The dapta platform currently provides access to a number OSS tools in the cloud, including CalculiX [6], OpenFOAM [7] and preCICE [6] as described below, but also OpenMDAO [8] for multidisciplinary optimisation applications, OpenVSP [9] for aircraft performance analysis and LibreOffice [10] for a wide range of scripting and reporting tasks. We aim to support users and developers of OSS software, by publishing OSS implementation templates, step-by-step tutorials and component source files. We are also committed to keeping OSS resources free of charge and easy to access.

The user manual tutorial 'A fluid-structure interaction simulation with preCICE, OpenFOAM and CalculiX' provides step-by-step instructions for a user to create and execute an FSI mutiphysics workflow using the dapta platform. OpenFOAM is used to model the fluid flow and CalculiX to model the behaviour of a flexible flap exposed to the fluid flow. We use the preCICE library [5] and the provided solver adapters to replicate the 'Perpendicular flap' preCICE tutorial in the cloud. The main elements of the preCICE library are shown in figure 5 and include: communication, data mapping, coupling schemes and time interpolation.

In general, preCICE participants are launched in separate processes and the coupled simulation only starts once both processes have been launched. To replicate this approach, we define a driver component that launches the fluid and solid components in parallel using the generic python ThreadPoolExecutor class. The FSI workflow therefore includes 3 components as shown in figure 3. We also need to create a connection that links the fluid component to solid component. It exists purely to satisfy the basic workflow requirements that simulation workflows should have a single starting component and that all (non-driver) components should be connected. In this case we choose the fluid component to be the start node (you can equally choose the solid component, without affecting the analysis in this case).

Once launched, the analysis completes within a few minutes and the user can download the

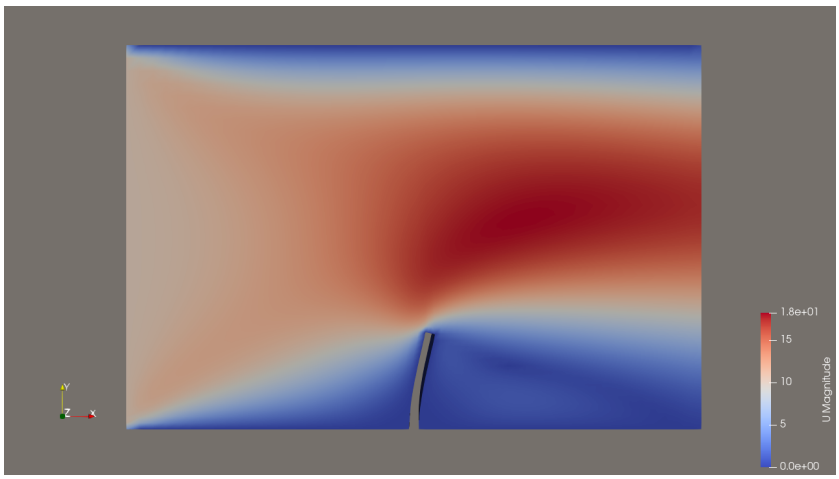

Figure 6: Plotting the FSI outputs: the fluid velocity field around the flexible flap in a channel flow

output files for further inspection. Figure 6 shows the openFOAM fluid velocity field around the flexible flap in a channel flow. The CalculiX output also includes the time trace of the flap tip deflections and applied fluid forces.

REFERENCES

[1] Big Compute 2021 State of Cloud HPC Report, *https://rescale.com/resources/big-compute-2021-state-of-cloud-hpc-report/*

[2] Gerd Schuhmacher, Ibrahim Murra, Liu Wang, Armin Laxander, Owen O'Leary and Michael Herold *Multidisciplinary Design Optimization of a Regional Aircraft Wing Box*. AIAA 2002-5406, 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, September 2002, *https://doi.org/10.2514/6.2002-5406*

[3] Lucian Iorga, Vincent Malmedy, Olivia Stodieck, Simon Coggon, Joseph Loxham *Preliminary Sizing Optimisation of Aircraft Structures - Industrial Challenges and Practices*. 6th Aircraft Structural Design Conference, Bristol, 20th September 2018

[4] Launching the Dapta Trial, *https://www.dapta.com/launching-the-dapta-trial/*

[5] Chourdakis G, Davis K, Rodenberg B et al. *preCICE v2: A sustainable and user-friendly coupling library [version 2; peer review: 2 approved]*. Open Res Europe 2022, 2:51, *https://doi.org/10.12688/openreseurope.14445.2*

[6] CalculiX website, *http://www.dhondt.de/*

[7] OpenFOAM website, *https://www.openfoam.com/*

[8] J. S. Gray, J. T. Hwang, J. R. R. A. Martins, K. T. Moore, and B. A. Naylor, *OpenMDAO: An Open-Source Framework for Multidisciplinary Design, Analysis, and Optimization*. Structural and Multidisciplinary Optimization, 2019.

[9] Robert A. McDonald and James R. Gloudemans, *Open Vehicle Sketch Pad: An Open Source Parametric Geometry and Analysis Tool for Conceptual Aircraft Design.* AIAA 2022-0004. AIAA SCITECH 2022 Forum. January 2022

[10] LibreOffice website, *https://www.libreoffice.org/*