Enabling Cloud-Based Computational Fluid Dynamics With a Platform-as-a-Service Solution

Simon J. E. Taylor, *Member, IEEE*, Anastasia Anagnostou, Tamas Kiss, Gabor Terstyanszky, Peter Kacsuk, Nicola Fantini, Djamel Lakehal, and Joris Costes

***Abstract*—Computational fluid dynamics (CFD) is widely used in manufacturing and engineering from product design to testing. CFD requires intensive computational power and typically needs high-performance computing to reduce potentially long experimentation times. Dedicated high-performance computing systems are often expensive for small-to-medium enterprises (SMEs). Cloud computing claims to enable low-cost access to high-performance computing without the need for capital investment. The cloud-based simulation platform for manufacturing and engineering simulation platform aims to provide a flexible and easy to use cloud-based platform-as-a-service (PaaS) technology that can enable SMEs to realize the benefits of high-performance computing. Our platform incorporates workflow management and multicloud implementation across various cloud resources. Here, we present the components of our technology and experiences in using it to create a cloud-based version of the Transport phenomena Analysis Tool CFD software. Three case studies favourably compare the performance of a local cluster and two different clouds and demonstrate the viability of our cloud-based approach.**

***Index Terms*—Cloud computing, computational fluid dynamics, fluid mechanics, high-performance computing, modeling and simulation.**

I. Introduction

COMPUTATIONAL fluid dynamics (CFD) uses numerical methods to analyze problems related to fluid or gas flows [1]. CFD is used in a wide range of engineering applications such as vehicle and aircraft engineering, biomedical engineering, thermo-hydraulic engineering, chemical engineering, and meteorology. CFD simulation has numerous advantages over physical testing in terms of experimental flexibility and cost-effectiveness. The heart of CFD simulation software is typically some kind of partial differential equation solver that uses complex algorithms to analyze the fluid/gas/solid flow. CFD also uses post-simulation tools such as analysis and visualization software to present the outcomes of the simulation.

The problems that CFD attempts to solve vary in complexity and size. In this context, we define complexity according to the number of phases in a simulation experiment. There are two types of problems, single phase and multiphase. Single-phase simulations involve the analysis of materials with the same phase, for example liquid–liquid, and multiphase simulations involve materials with different phases or different chemical properties, for example, liquid–gas, oil–water, etc. The latter presents higher levels of complexity. The size of the problem relates to the resolution of the experiment. As mentioned above, CFD solves partial differential equations in order to obtain results. With the exception of very simple CFD problems, the partial differential equations cannot be solved analytically. Therefore, the flow area is split in subdomains (i.e., geometric primitives) and the governing equations are solved within each subdomain. These subdomains are called cells or elements and the collection of all cells is called a grid or a mesh. Typically, with larger meshes we can obtain greater accuracy. However, as the resolution increases, the computational requirements to solve the partial differential equations also increase. This can significantly increase the time taken to analyze a problem.

Advances in high-performance computing have had a significant impact on CFD. Parallelization strategies using threading, message passing interface (MPI) or parallel virtual machine approaches allow for more accurate, faster CFD analysis in a smaller time scale [2], [3]. A mesh is mapped to many different CPUs so that the partial differential equations can be solved in parallel supported by tightly coupled high-speed communication. Companies with access to high-speed computing clusters can, therefore, expect faster CFD analysis and the opportunity to create better designs and solutions [4]. There are many examples where approaches to CFD parallelization have been investigated, particularly using MPI [5]. For example, Shang [6] discusses the parallelization performance of different meshing methods by using Code Saturne, a CFD software with MPI

Manuscript received May 19, 2017; revised March 8, 2018; accepted June 2, 2018. Date of publication July 2, 2018; date of current version January 3, 2019. This work was supported by the Cloud-based Simulation platform for Manufacturing and Engineering (CloudSME) under Grant 608886 (FP7-2013-NMO-ICT-FOF). Paper no. TII-17-1067.
(Corresponding author: Simon J. E. Taylor.)

S. J. E. Taylor and A. Anagnostou are with the Modelling and Simulation Group, Department of Computer Science, Brunel University London, Uxbridge UB8 3PH, U.K. (e-mail: simon.taylor@brunel.ac.uk; anastasia.anagnostou@brunel.ac.uk).

T. Kiss and G. Terstyanszky are with the Centre of Parallel Computing, University of Westminster, London W1W 6UW, U.K. (e-mail: T.Kiss@westminster.ac.uk; G.Z.Terstyanszky@westminster.ac.uk).

P. Kacsuk is with MTA SZTAKI, Budapest 1132, HUNGARY (e-mail: kacsuk.peter@sztaki.mta.hu).

N. Fantini is with CloudBroker GmbH, Zurich 8005, Switzerland (e-mail: nicola.fantini@cloudbroker.com).

D. Lakehal is with Ascomp GmbH, Zurich 8005, Switzerland (e-mail: lakehal@ascomp.ch).

J. Costes is with CMLA, ENS Paris-Saclay and CNRS, Cachan 94235, France (e-mail: joris.costes@cmla.ens-cachan.fr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2018.2849558

parallelization functions and concludes that the mesh partitioning method affects the performance of the parallelized simulation both in terms of latency and efficiency. The behavior of different partitioning approaches impacts performance. Multicore computing facilities, such as those provided by super computer centers are being used to support the processing of huge CFD applications [7]. Generally, it is likely that a company having access to their own high-performance computing facilities will be able to process CFD simulations faster. This is an issue for small-to-medium enterprises (SMEs) as the cost of purchasing and maintaining their own computing cluster may be prohibitive, and therefore, makes it difficult to compete with larger companies with access to this technology.

Cloud computing offers widely available, easily accessible, on-demand computing resources that can be rapidly accessed to support a specific task and then discarded [8]. Generally, this model of service provision enables access to hardware without capital outlay or the on-going cost of IT management, and access to software potentially on a pay-per-use basis rather than an annual license. In terms of high-performance computing, cloud gives the opportunity to hire large amounts of computing resources on an economical basis. However, the development of cloud-based solutions is not a simple task due to the variety and choice of technologies that must be accommodated. For example, cloud computing offers at least four different deployment models and provides services at varied levels of abstraction as described by cloud service models. These include cloud provisioning that is private within the boundaries of an organization, community clouds used by organizations with common interests, public clouds that are accessible by public networks, or hybrid clouds that can combine aspects of all three. The cloud service model, or the cloud stack, includes infrastructure-as-a-service (IaaS—management and hosting of physical cloud elements such as computing, networking, etc.; deployment of these as IT infrastructure according to the above cloud deployment models), platform-as-a-service (PaaS—provision and management of the deployed cloud infrastructure as well as middleware to support the deployment and development of cloud-based applications), and software-as-a-service (SaaS—the actual software application deployed on a cloud platform that uses the cloud infrastructure resources).

As noted by Runchal [9], from a CFD simulation software perspective, cloud computing is very attractive as it could support the creation of a new generation of high speed, pay-per-use CFD applications that are composed of interoperating sets of simulation software services. For example, the data for a CFD experiment can be stored in a cloud-based database that is connected to a cloud-based CFD solver. The simulation is run on multi-CPU instances that support the processing requirements of parallelized CFD. Results can be analyzed and visualized by a cloud-based post-simulation tool. He also supports the view that for those who do not have access to these facilities, cloud computing has the potential for cost-effective access to CFD applications that utilize such architectures, though on a smaller scale. Evangelinos and Hill [10] used on-demand EC2 clusters to test the performance of cloud-based clusters and concluded that, at the time, cloud-based high-performance computing could be compared with low cost in-house clusters. Ledyayev and Richter [11] report on a high-performance computing cloud implementation of OpenFOAM that gave best results when using a single cloud-based virtual machine with multiple cores. O'Leary *et al.* [12] ported the Hydra-TH CFD simulation onto their open source high-performance computing/cloud architecture using a platform approach implemented on the Amazon EC2 cloud. Their results indicate comparable performance with local noncloud resources. The importance of mapping MPI processes onto cores to reflect intensive communication arising from model structure has been recognized by Guzzetti *et al.* [13]. They also identify that clouds could be a cost-effective resource for CFD and make available powerful hardware configurations, especially when taking mapping into consideration. These examples show that cloud is a feasible cost-effective technology that could be used to speed up CFD simulations. Indeed there are examples in industry where CFD simulations are executed on cloud (e.g., Sabalcore, Rescale, Ciespace, Autodesk A360, etc.) However, without an appropriate cloud computing skill-base, SMEs creating such applications face a costly and steep learning curve and, with the exception of the Hydra-TH example, there is little evidence of cloud-based platforms that could be used to simplify and reduce the cost of development. Further, most of the above examples use a single cloud provider (usually Amazon). Different cloud providers charge different rates for their instance types and the quality of service can vary. It would be, therefore, very useful to easily switch between cloud providers without being locked in to a single provider. However, cloud providers' platform technologies vary and require work to port an application from one cloud to another. Finally, any commercial cloud product or service requires some kind of charging mechanism that would allow users to be charged for their use of the application and cloud. None of the above examples have these functionalities.

SME developers of high-performance CFD simulations therefore need technology that will allow them to quickly and economically develop multicloud-based applications that can use multi-CPU instances as appropriate. In addition to the above, other simulation application types (e.g., discrete process simulation) need high-performance computing that can run multiple simulations simultaneously to speed up experimentation. A cloud platform supporting these requirements for high-performance simulation would, therefore, have to: 1) to allow parallelized applications to run on multiple, tightly coupled CPUs available on a single cloud instance and 2) to run multiple simulations in parallel on a combination of multiple single cloud instances (e.g., as if the experiments were running on multiple computers). It would, therefore, need to deploy onto a variety of clouds and instance sizes, and these jobs must be able to run in parallel on multiple sets of cloud instances. The commercial requirements of the SMEs mean that the platform must also be able to charge for use. To attempt to develop a platform to deliver cloud-based high-performance simulation for SMEs, the cloud-based simulation platform for manufacturing and engineering (CloudSME) project (www.cloudsme-apps.com) has developed the CloudSME simulation platform. To satisfy the above requirements we selected 1) CloudBroker (www.cloudbroker.com) to satisfy the requirement of multicloud/multiinstance deployment as it provides a common application management interface to different clouds that allows applications to be easily deployed and switched from one cloud to another (i.e., the cloud specific interfacing and deployment has already been implemented in CloudBroker, and therefore, saves a developer from having to do it themselves); 2) the WS-PGRADE/gUSE [14] science gateway framework to implement a workflow system that allows multiple experiments (and other tasks) to be launched and managed; and 3) an AppCenter developed within the project to allow a company to charge for use of its application on cloud. Other technologies in this area have some multicloud functionality and have evolved from e-Science and Grid workflow computing platforms [15]. For example, Wang *et al.* [16] show how KEPLER can be integrated with federated cloud resources via CometCloud; Juve *et al.* [17] describe an extension of the PEGASUS workflow system over three clouds and report comparable performance between cloud and grid implementations; *Zhao et al.* [18] present an approach to running SWIFT on cloud. These examples have been developed for scientific applications and do not appear to have the mechanisms to support commercial requirements such as billing and payment. Similarly, a limited number of multicloud platforms exist (e.g., RightScale and Scalr) but do not have workflow integration.

To illustrate how our paper can be used to create high-performance CFD simulations, we present two contributions in this paper: the CloudSME simulation platform and how it can be used to develop a cloud-based CFD application. The cloud-based application is ASCOMP's TransAT (Transport phenomena Analysis Tool) CFD tool. We have chosen TransAT to illustrate our approach as it is a commercial tool used to simulate a wide range of single and multifluid/component flows with heat transfer. Typical applications of TransAT range from surface-tension dominated flows (e.g., microfluidics systems) to large-scale turbulent flows (e.g., hydrodynamics of ships and submarines) across a range of industries. Section II gives more information on the CloudSME simulation platform. Section III presents our experiences in developing the cloud-based CFD simulation. Section IV presents three case studies to show the successful use of cloud. Section V discusses the benefits and limitations of this approach and Section VI summarizes the main contributions and future work.

II. CloudSME Simulation Platform Architecture

Fig. 1 shows the CloudSME simulation platform architecture. The AppCenter and platform application programming interfaces (APIs) form the simulation applications layer, WS-PGRADE/gUSE and CloudBroker form the cloud platform layer and the various cloud infrastructures that the platform and applications access are made available in the cloud resources layer. Conceptually, to use the platform software requiring cloud deployment and/or cloud-based high-performance computing is redeveloped as a software-as-a-service at the simulation applications layer, hosted by the cloud platform layer (platform-as-a-service) and uses cloud resources made available through the cloud resources layer. The CloudSME simulation platform has been used to implement 11 commercial simulation products across a wide range of industrial applications (see http://www.cloudsme-apps.com/practical-examples/ for examples of CFD, process simulation, and computer aided design). The technical details of the integration of the platform's technologies are presented in [19]. We now describe the functionality of the components of the Platform.

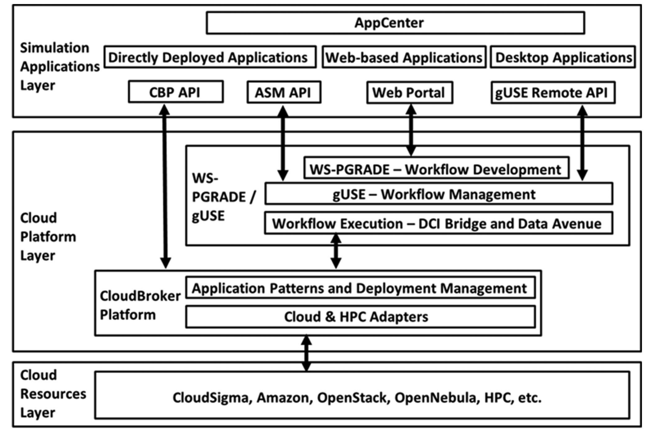


Fig. 1. CloudSME simulation platform architecture.

A. AppCenter

This is a generic entry point to browse and execute various simulation applications. The AppCenter supports three main development options: directly deployed applications, desktop applications, and web-based applications. Directly deployed applications deploy and offer cloud applications directly from the AppCenter. The software provider registers and publishes the application in the CloudSME AppCenter. End-users can then access and execute these applications directly from the AppCenter and are charged for their utilization based on software provider defined pricing policies (e.g., subscription-based pricing, resource consumption-based pricing, fixed charge for each simulation run, or even free of charge execution). The advantage of this solution is that the AppCenter provides easily customizable options to set up various pricing models and it is also responsible for accounting and billing. Users simply need to add credit to their account and can then execute simulation applications on all cloud resources supported by the selected application. The complexity of a multicloud execution and billing is handled by the platform. Desktop applications extend desktop software with cloud execution support using one of the platform APIs to redirect computation intensive simulations to cloud. Users download the package from the AppCenter, install it on their own machine, and then utilize its cloud extended capabilities. This solution enables vendors to keep the user interface (with minor modifications) of their original product and to use the accounting and billing functionality of the AppCenter accessed through their product. Web-based applications are simply linked to the AppCenter. The web application can provide a complete custom portal-based user interface and can utilize capabilities of the platform and the AppCenter accounting and billing mechanisms via suitable APIs. To enable the development of applications that are directly deployed in the AppCenter or the extension of desktop applications with cloud support either the CloudBroker Java Client Library API or REST API, or the WS-PGRADE/gUSE Remote API can be used. Using the CloudBroker APIs bypasses WS-PGRADE/gUSE and provides direct access from the application to the multicloud resources supported by the CloudBroker (i.e., when workflow is not required). Using the Remote API of WS-PGRADE/gUSE enables developers to execute complex application workflows linking multiple application components together on multiple clouds. For web-based applications, either the application specific module (ASM) API of WS-PGRADE/gUSE is used that enables the rapid development of a custom portal/gateway in the form of customised Liferay Portlets, or a completely custom web interface can be developed embedding either CloudBroker API or gUSE Remote API calls. Alternatively the web-based front end to WS-PGRADE/gUSE or CloudBroker can also be applied to launch workflows or cloud applications as appropriate.

B. WS-PGRADE/gUSE

WS-PGRADE/gUSE supports the development and deployment of high-performance computing applications across different types of distributed computing infrastructures such as clusters, desktop grids, and clouds. It is widely used to create high-performance computing applications by a range of scientific communities. WS-PGRADE/gUSE consists of three tiers: a presentation tier, a middle tier, and an architectural tier. The presentation tier consists of Web Services Parallel Grid Runtime and Developer Environment Portal (WS-PGRADE). It is implemented in the Liferay web portal framework (www.liferay.com) and has a graphical workflow editor that allows users to create and populate workflows for their applications that run on various distributed computing infrastructures (including clouds). Once a workflow is created it is saved and managed in the middle tier gUSE services. The gUSE services provide a complete environment for workflow support and execution. It facilitates the management, storage, and execution of workflows and has flexible deployment options (single hosting or distributed hosting to optimize resource usage or increase performance). It also offers various data services, job management services, and workflow discovery services. It provides secure access and authorization to distributed computing infrastructure resources by using appropriate security certification (the CloudSME simulation platform implements cloud security via CloudBroker). In the architectural tier, gUSE uses the distributed computing infrastructure bridge (DCI Bridge) job submission service to access and manage the resources of various distributed computing infrastructures. There are many different classes of distributed computing infrastructure, each with their own complex management requirements. To create a simplified and layered approach to cloud development, clouds from the perspective of WS-PGRADE/gUSE, are treated as a single class of distributed computing infrastructure. In the CloudSME simulation platform, the responsibility for the management of different clouds is given, therefore, to CloudBroker. For example, when a workflow task is processed in gUSE, the DCI Bridge is used to submit that task to CloudBroker. The platform then manages that task on the specified cloud.

C. CloudBroker

A typical cloud implementation involves software being developed for one cloud (e.g., Amazon). Different clouds offer different functionality and costs. The concept behind the CloudBroker part of the CloudSME simulation platform is that it presents a cloud platform that developers can use to manage their implementations across multiple clouds. This means that once an application has been deployed on different clouds, CloudBroker can help developers deliver a flexible service that enables cloud-based applications to be easily managed (and charged) across product offerings based on different cloud infrastructures. The CloudBroker environment is a web-based application store for the deployment and execution of compute-intensive applications on a cloud and widely automates user, software, resource, job, and invoice management. It is suitable for any kind of batch-oriented command line software, both Linux- and Windows-based, and both serial or parallel processing (via MPI for example). It can be accessed through any web browser and through different APIs.

To help developers create cloud-based applications across a variety of clouds, CloudBroker implements a range of cloud adapters. Once a cloud adapter has been implemented for a given cloud, CloudBroker effectively provides a common API for users to develop their applications (and so reduce the time taken to learn how to deploy to a different cloud). A user, therefore, develops their application for the CloudBroker APIs to create an application pattern for their software and cloud variant (typically distinguished by operating system requirements rather than specific cloud middleware). This means that once a user has successfully created an application pattern on a cloud, development time for the next and subsequent clouds is shorter. The range of adapters currently includes Amazon web services, CloudSigma, Open-Stack, OpenNebula, and Eucalyptus. New ones are being developed for high-performance computing centers including CINECA in Italy, HLRS in Germany, and Romeo in France. CloudBroker handles requests to instantiate a compute instance on a given cloud by using the adapter to interact with that cloud. Run-time process monitoring, queueing, resource, storage, and image management services deal with requests to run specific software on specific cloud instances. The platform also provides process, user and application management, accounting, billing and payment modules, as well as security and fault tolerance management. A public version of CloudBroker can be accessed at platform.cloudbroker.com.

III. Cloud-Based TransAT

ASCOMP's TransAT consists of a GUI (front end) and a CFD solver (back end). The front end is used to setup the simulation, to define a mesh decomposition and to preprocess the required input parameters. This is then passed to the back end which compiles and then executes the simulation using the CFD solver. Once the simulation is finished, the results are visualized using postprocessing software such as Paraview (www.paraview.org) or Tecplot (www.tecplot.com). The back end simulation execution can take a few seconds or a few weeks depending on the complexity of the experiment and computational capabilities. TransAT implements high-performance computing parallel execution by using Open-MPI (www.open-mpi.org). From the TransAT GUI, a user can specify how many CPUs of a compute cluster to use. TransAT then manages the decomposition of the simulation across the available CPUs. End users, therefore, require their own cluster or need to hire ASCOMP's cluster to exploit the parallel implementation of TransAT. Discussions with ASCOMP on the development of a cloud-based version of TransAT identified two main requirements: how could users use scalable high-performance computing and TransAT without access to expensive computing hardware, and how could the tool be more widely distributed? Cloud, therefore, presented an attractive option as hardware could be hired on-demand and the tool could be potentially accessed and distributed via the web. It was hoped that the closely coupled CPUs of single cloud instances would also deliver a scalability solution (i.e., larger instance sizes should deliver faster performance as TransAT is already parallelized).

The three development options were considered. Direct deployment into the AppCenter and web-based implementations would have required a significant redevelopment of TransAT's front end into a web-based version. The desktop alternative was, therefore, more attractive as only minor modifications were required. Setting up a download area on the AppCenter meant that users could download the package from the web-based AppCenter, install it on their own machine, and then run their simulations on cloud through the TransAT front end. Prior to running, a user would set up an account on the AppCenter. The user would then login into TransAT and run his/her simulation. TransAT would synchronize with the AppCenter to bill the user for cloud use. No workflow would be required as TransAT only ran one simulation at a time. The requirements for cloud-based TransAT were, therefore, to allow a user to select the cloud/cloud instance (and region if required), to manage the execution of the simulation on cloud, and to bill for the use of cloud. Fig. 2 shows the architecture and flow diagram of the cloud-based tool. As can be seen, the architecture consists of three elements: the TransAT GUI installed in a user workstation with a visualization tool, the deployment on the CloudSME simulation platform cloud platform layer and the cloud resources layer on which TransAT runs and stores its output (note the AppCenter is not shown). The CloudBroker REST API was chosen to develop the application (the Java API was not appropriate for this implementation) [20].

The following steps were followed to create the cloud-based application. First, an Application Pattern was created to enable the deployment of the TransAT back end on cloud. This consisted of an installation package containing an installation shell script, the zipped TransAT solver, and a licence key. The TransAT Application Pattern uploaded to CloudBroker using the CloudBroker web-based system. TransAT's front end was then modified to allow users to login into the AppCenter, to select from a list of available clouds and cloud instances (uploaded from CloudBroker at runtime) and to run and monitor their simulation. An AppCenter entry was created for TransAT that allowed users to create an account (and add credit) and download TransAT. To use cloud-based TransAT, a user would, therefore, create an account and download and install TransAT. The user then creates the CFD model, logs into his/her account to identify themselves, selects a cloud/cloud instance and then runs the model. The run instruction sends the model with cloud details to CloudBroker as an XML job description. On receipt, CloudBroker synchronizes the user's account with the AppCenter and then creates the job. This involves taking the appropriate TransAT Application Pattern for the requested cloud/instance, invoking a specific cloud instance on the selected cloud and then uploading and executing the Application Pattern on that instance (TransAT's back end involves the compilation of TransAT tools over the CPUs of the instance and the authentication of the licence key). When complete, CloudBroker then uploads the model to the cloud instance and instructs TransAT to begin the simulation. The TransAT front end periodically polls CloudBroker to determine if the run has ended. CloudBroker then synchronizes the cost of the run with the AppCenter. The user can then download the results.

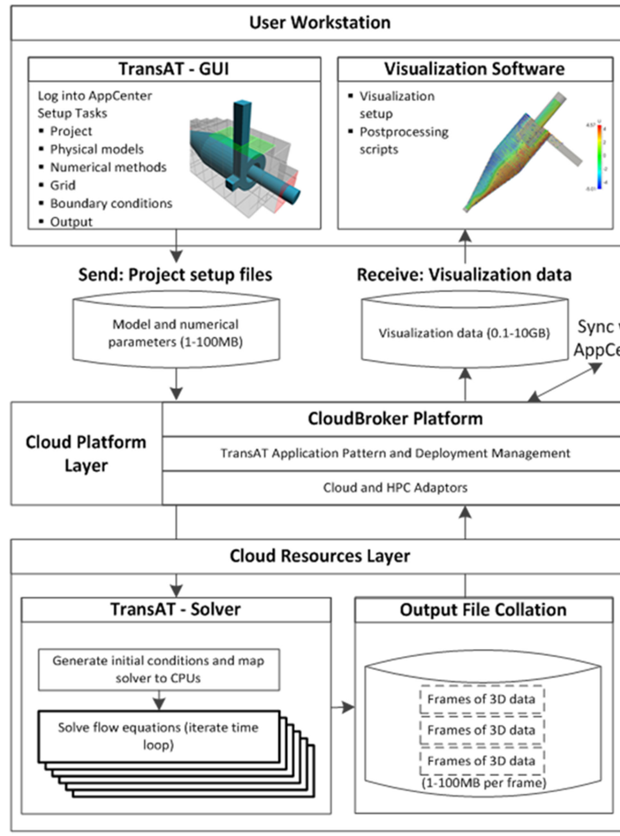


Fig. 2. Cloud-based TransAT architecture and flow diagram.

IV. Case Studies

We now present three case studies showing the use and performance of cloud-based TransAT. The aim of these case studies is not to present a comprehensive performance test across every available cloud accessible through the CloudSME simulation platform but to present examples that illustrate the benefit of our approach to CFD simulation. We restrict our examples to those that should be familiar to practitioners in the oil and gas industries (major users of CFD). During field production, the extracted product from a well naturally contains oil and water. Water also can be injected during the process in order to force the oil to surface. Different techniques are used for separation,

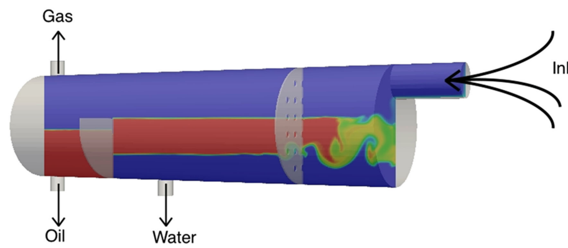


Fig. 3. TransAT visualization of a gravity-based horizontal three phase separator.

one of which is the gravity-based horizontal three-phase separator. The device is a cylindrical vessel that, given a sufficient residence time, allows separation of water, oil, and gas with the help of gravity. It is very important for the oil industry to utilize the separators efficiently. The first case study illustrates how cloud-based simulation can speed up simulation in this important area. The second case study shows how cloud can be used to speed up the simulation of a well-known associated numerical benchmark. This was proposed by the company Gaztransport & Technigaz (GTT) during the 2009 International Society of Offshore and Polar Engineers (ISOPE) [21] based on the impact of liquid inside liquid natural gas (LNG) carriers due to sloshing (i.e., a liquid surrounded by a gas in a closed tank). Gas and liquid are initially at rest, their densities are constant and pressure is homogeneous. In the initial state, liquid starts to fall freely under action of gravity. Eventually, liquid impacts on the rigid horizontal tank base. The third case study in capillary action represents another common simulation problem in this area. Capillary action is the ability of a liquid to flow in narrow spaces without, or against, an external force. This phenomenon occurs at the interfaces between two immiscible fluids or between a fluid and a surface. This case was also selected as capillary action has an important role in micro structures in several industrial and medical applications, such as micro heat exchangers, lab-on-chips, bio-MEMS and micro cooling electronics [22]. Simulation is needed to optimize and develop the design of such devices.

A. Gravity-Based Horizontal Three-Phase Separator

The device that was simulated has one inlet for the field product and three outlets for the oil, water, and gas separated products. Fig. 3 shows a TransAT visualization of the separator. To enhance the separator performance, several inlet devices are in place. For example, the inlet momentum breaker is used in the first phase to reduce the momentum of the input stream. Within the cylinder there are mechanical devices that reduce fluid velocity and allow the liquids to drop in the accumulation sections. The mesh size of the selected simulation includes 300 000 cells and the three phases of the device increase the model's complexity.

We conducted experiments on ASCOMP's local high-performance computing cluster and on cloud resources using the CloudSME simulation platform. The runs were performed using up to 32 processors on the local cluster (Intel Xeon E5649 2.53 GHz processors with 32 GB memory). On cloud, for experiments up to 8 CPUs, general purpose balanced Amazon EC2 were used via the CloudSME simulation platform [all Intel Xeon E5-2670 2.50 GHz CPUs with sizes large (2 CPU with 7.5 GB memory), xlarge (4 CPU with 15 GB memory) and 2xlarge (8 CPUs with 30 GB memory)]. For the experiments with 16 and 32 CPUs, compute optimized Amazon EC2 instances were used [all Intel Xeon E5-2680 2.80 GHz CPUs: 4xlarge (16 CPU with 32 GB memory), 8xlarge (32 CPU with 32 GB memory)]. The results in Table I show the runtimes in seconds of the gravity separator CFD model executing 3000 iterations on different instances.

TABLE I
RUNTIME IN SECONDS (3 000 ITERATIONS)

Number of processors	2	4	8	16	32
Local cluster	412,319	153,687	121,791	78,805	38,667
Amazon EC2	678,000	358,500	198,000	99,000	57,000

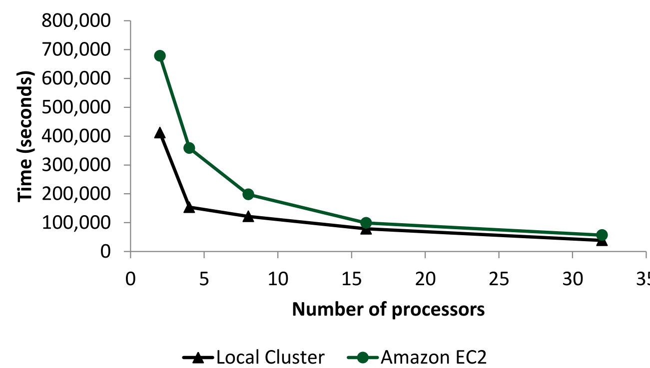


Fig. 4. Gravity separator simulation runtime (3 000 iterations).

Fig. 4 shows the runtime of the local cluster versus the balanced instances of Amazon EC2. Both follow a similar trend. Comparing runtimes between 2 and 8 CPUs, the cluster improved by a factor of 3.39 and the cloud by 3.42 on comparable hardware. As would be expected, the simulation ran slower on the cloud instances than the cluster by a factor of 0.60, 0.42, 0.60, 0.80, and 0.68 for 2, 4, 8, 16, and 32 CPUs, respectively. The cluster hardware is a better specification than the EC2 instances and the comparative variance reflects the faster communication speed between CPUs. Overall, in spite of the fact that the cloud-based platform seems to be slower than the local cluster when one looks at the clock time, the cloud performance is actually very good in that it reproduces comparable speed-up to the cluster. This is exactly the objective of the entire concept: that is to offer the possibility for SMEs to reproduce the same high-performance computing performance expected from a local (expensive to buy and maintain) cluster on a cloud-based platform.

As shown in Table II, the comparative cost of cloud computing makes a compelling argument. For example, at the time of writing the cost of the largest instance (8xlarge) is \$1.68/hr. The time taken to run the simulation on this instance is 57 000 s (15.83 hr). The cost to run this simulation on that instance size is therefore \$26.60. The cost of the cluster used in this experimentation (a 64 processor rack with high-speed interconnection) is approximately \$24,000 (not including maintenance costs and the considerable energy consumption costs for power and

TABLE II
AMAZON EC2 COST

	Number of CPUs	Cost per hour ($)	Runtime (sec)	Total cost ($)
Amazon large	2	0.14	678,000	26.37
Amazon xlarge	4	0.28	358,500	27.88
Amazon 2xlarge	8	0.56	198,000	30.80
Amazon 4xlarge	16	0.84	99,000	23.10
Amazon 8xlarge	32	1.68	57,000	26.60

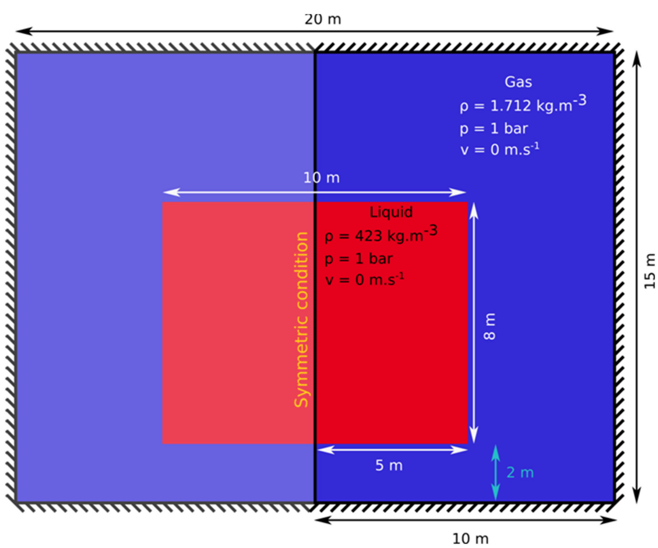


Fig. 5. Two-dimensional liquid natural gas tank ISOPE benchmark.

cooling). At a cost of under $40 this cloud performance is arguably acceptable at the cost and well within any SME's budget.

B. GTT Benchmark

The second case study was performed by Eurobios as an independent demonstration. Eurobios is a consultancy that uses TransAT to simulate various problems for their clients in the oil and gas industries including natural convection, gravity separation of a mixture, vortex/turbulence problems, and bubble merger problems.

The interface between liquid and gas is described using the ensemble average method developed in TransAT (this uses an average of the properties of the fluids contained in a mesh cell of the mesh). Boundaries are modeled as rigid walls. The numerical scheme used is a finite volume implicit scheme in which in a given cell of the mesh, quantities are approximated using the volume integral of the cell quantities; the evolution of the quantities is computed using the fluxes through the edges of the cell (a flux represents the contribution of a given cell on the evolution of the state of its neighbor). As shown in Fig. 5 the parameters are: Liquid density 423 kg. m^{-3}, initial fluid velocity 0 m.s^{-1}, pressure 1 bar; Gas density 1.712 kg.m^{-3}, initial fluid velocity 0 m.s^{-1}, pressure 1 bar. The dimensions of the cell are indicated in the figure.

Two instance types were used in performance testing: 1) Amazon 16 cores, EC2 Compute Optimized—4xlarge, 30 GB RAM (0.93EURO/hour 0.14EURO/GB) and 2) CloudSigma 24 cores, 60 GHz, 32 GB RAM (price withheld).

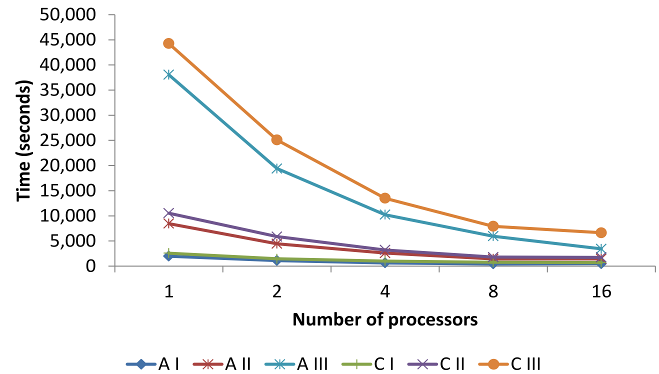


Fig. 6. GTT runtime comparison.

TABLE III
GTT RUNTIME IN SECONDS.

CPUs	1	2	4	8	16
A I	1,980	1,116	651	444	503
A II	8,460	4,446	2,601	1,445	1,456
A III	38,049	19,391	10,231	5,956	3,435
C I	2,578	1,472	1,020	779	709
C II	10,526	5,869	3,226	1,803	1,757
C III	44,257	25,092	13,512	7,920	6,649

TABLE IV
GTT SPEEDUP

CPUs	2	4	8	16
A I	1.77	3.04	4.46	3.94
A II	1.90	3.25	5.85	5.81
A III	1.96	3.72	6.39	11.08
C I	1.75	2.53	3.31	3.64
C II	1.79	3.26	5.84	5.99
C III	1.76	3.28	5.59	6.66

In performance testing, three regular meshes were considered: 15 000 cells, 60 000 cells, and 240 000 cells on Amazon and CloudSigma (experiments A I, A II, A III, C I, C II, C III, respectively). Fig. 6 and Table III show the runtime in seconds for each of the above experiments.

Table IV shows the speedup. As can be seen performance ranged between 1.75 and 1.96 for 2 CPUs (average: A 1.88, C 1.77), 2.53 to 3.72 for 4 CPUs (average: A 3.34, C 3.02), 3.31 to 6.39 for 8 CPUs (average: A 5.57, C 4.91), and 3.64 to 11.08 for 16 CPUs (average: A 6.94, C 5.43). The average performance indicates moderate speedup. The best performance was obtained for the most refined meshes reflecting a better ratio of computation to communication for each node. Overall Amazon performed marginally better than CloudSigma with the exception of the 16 CPU instance where speedup was significantly higher (A 11.08, C 6.66). This might reflect the compute optimized nature of the Amazon instance. The cost differential between the clouds was not available at the time of writing. Again this shows good comparative utilization.

C. Capillary Movement

The third case study was also conducted by Eurobios. In this demonstration, a two-dimensional (2-D) axis-symmetric CFD simulation has been performed that represents the movement of

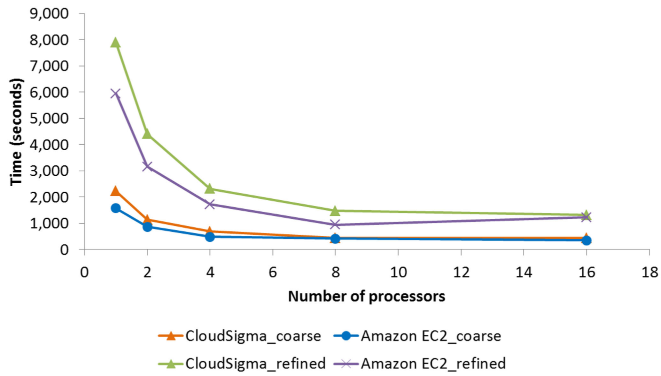


Fig. 7. Capillary movement runtime comparison.

TABLE V
CAPILLARY MOVEMENT SPEEDUP

CPUs	2	4	8	16
Amazon coarse	1.82	3.22	3.80	4.45
CloudSigma coarse	1.95	3.21	4.98	5.00
Amazon refined	1.89	3.46	6.28	4.82
CloudSigma refined	1.79	3.40	5.33	5.98

water up a tube filled with air. The computation domain is equal to 1 cm in the horizontal direction and to 1.5 cm in the vertical direction. Two uniform meshes were run containing 5 400 cells (coarse) and 21 600 cells (refined), respectively. The instance types used in the experimentation were the same as the GTT case study. Fig. 7 shows the runtime in seconds for each of the above experiments.

Table V shows the speedup. As can be seen performance ranged between 1.79 and 1.95 for 2 CPUs (average: A 1.86, C 1.87), 3.21 and 3.46 for 4 CPUs (average: A 3.34, C 3.31), 3.80 and 6.28 for 8 CPUs (average: A 5.04, C 5.16), and 4.45 and 5.98 for 16 CPUs (average: A 4.64, C 5.49). The average performance again indicates moderate speedup. As with the GTT case study, the best performance was obtained for the most refined meshes and overall Amazon performed marginally better than CloudSigma with the exception of the 16 CPU instance where CloudSigma outperforms Amazon (A 4.82, C 5.98). The cost differential between the clouds was not available at the time of writing. Good comparative utilization is again demonstrated.

V. DISCUSSION

As discussed, the above case studies are illustrative of many different CFD applications. Cited literature identified that mesh partitioning affects the performance, cloud computing has the potential for cost-effective access to CFD applications that utilize high-performance computing architectures (though on a smaller scale), and that cloud implementation could be compared with local noncloud resources. Our case study results agree with these observations to some extent. With larger cloud instances case study A shows comparable run times with a local cluster. Case studies B and C show better performance with larger clusters and the difference in performance between cloud providers. Both case studies also show the impact of cell density on performance with higher density simulations benefitting from larger cloud instances. Case study C demonstrates that this may be limited with some simulations with little extra benefit beyond 8 CPU instances. This may well be due to the processing requirements of smaller models. However, the overall cost of the simulations is significantly lower than purchasing a local cluster. Overall, the use of our Platform to produce cloud-based TransAT means that SMEs can access faster CFD simulation without the need for costly hardware investment. As cloud providers regularly update their hardware, this also means that SMEs can also access contemporary computing systems. Further work is required to understand the relationship between simulations, mesh densities, and the cost of ideal cloud instance sizes. This is the next step in our development and will use the Amazon EC2 C3 and C4 enhanced networking instances, spot instances, and similar offerings by other providers available through CloudBroker. Other clouds such as those provided through high-performance computing centers are also being tested.

In terms of service development and delivery, the experience of ASCOMP with the CloudSME simulation platform was very positive. Initially both developers and the CloudSME simulation platform technology providers had to understand the best design strategy. Once the approach described above had been understood, it was relatively straightforward to port TransAT to CloudBroker for deployment on Amazon and CloudSigma clouds. The entry in the AppCenter and modification to TransAT's user interface means that ASCOMP can easily charge for cloud use. The CloudSME simulation platform, therefore, enabled this menu-driven choice over which cloud type and instance size can be selected easily for a CFD simulation. It also reduced significantly the product development time to around two months and removed the need to learn how to deploy TransAT to different clouds. Indeed to add new clouds and cloud instances to TransAT involves updating the application pattern and registering it on CloudBroker (rather than learning how to use the new cloud technology directly). In terms of further development, running the CFD simulations with different parameters is currently done manually. However, WS-PGRADE/gUSE will make it possible to automate this by using dedicated workflows exploring a variety of parameter sets. Workflows for CFD simulation experimentation are, therefore, being investigated. Other cloud-based products and services have been deployed on the platform and are being used commercially. For example, experiences in developing the cloud-based 3-D scan insole designer for tailored shoe insoles using the web-based application development approach are reported in [23].

The multicloud approach is rapidly evolving. Grozev and Buyya [24] and Toosi *et al.* [25] discuss the evolution of multiple clouds into a federated architecture or Intercloud. The benefits of an Intercloud include diverse geographical locations (impact on performance and legislative requirements), better application resilience (fault tolerance), and avoidance of vendor lock. The IEEE Intercloud Testbed project [26] has been created to facilitate the development of Intercloud. Both multicloud and Intercloud approaches could take advantage of tools such as Cloud Crawler [27] and CLAudit [28] that enable cloud performance monitoring over time.

VI. Conclusion and Further Work

The CloudSME simulation platform was created to help SMEs to develop commercial multicloud-based simulation applications. Although elements of the CloudSME simulation platform exist, as far as we are aware, there is no single PaaS approach that brings together workflow services and multicloud deployment for commercial applications. We have presented the two contributions of this paper: a novel platform that has been used to enable the rapid development of cloud-based simulation applications and a practical demonstration of how this can be applied to CFD simulation.

The commercial release of cloud-based TransAT is currently underway and is now available at www.cloudsmeapps.com/production-appcenter along with many other cloud-based industrial applications enabled by the CloudSME simulation platform PaaS. The GTT model is available from 10.5281/zenodo.1189315. Further work will investigate the relative performance of cloud-based TransAT using various clouds as noted in the discussion section and the impact of using workflows to automate experimentation.

Acknowledgment

The authors would like to thank Dr. S. Reboux for his contributions to this paper.

References

[1] J. D. Anderson, *Computational Fluid Dynamics: The Basics with Applications*. New York, NY, USA: McGraw-Hill, 1995.
[2] G. A. Geist, J. A. Kohl, and P. M. Papadopoulos, “PVM and MPI: A comparison of features,” *Calculateurs Paralleles*, vol. 8, no. 2, pp. 137–150, Jun. 1996.
[3] R. Jia and B. Sunden, “Parallelization of a multi-blocked CFD code via three strategies for fluid flow and heat transfer analysis,” *Comput. Fluids*, vol. 33, no. 1, pp. 57–80, Jan. 2004.
[4] S. Cant, “High-performance computing in computational fluid dynamics: Progress and challenges,” *Philos. Trans. A Math. Phys. Eng. Sci.*, vol. 360, no. 1795, pp. 1211–1225, Jun. 2002.
[5] A. Afzal, Z. Ansari, A. R. Faizabadi, and M. K. Ramis, “Parallelization strategies for computational fluid dynamics software: State of the art review,” *Arch. Comput. Methods Eng.*, vol. 24, no. 2, pp. 337–363, Apr. 2016.
[6] Z. Shang, “Impact of mesh partitioning methods in CFD for large scale parallel computing,” *Comput. Fluids*, vol. 103, pp. 1–5, Nov. 2014.
[7] R. Borrell, J. Chiva, O. Lehmkuhl, G. Oyarzun, I. Rodrguez, and A. Oliva, “Optimising the Termofluids CFD code for petascale simulations,” *Int. J. Comput. Fluid Dyn.*, vol. 30, no. 6, pp. 425–430, Sep 2016.
[8] P. Mell and T. Grance, “The NIST Definition of Cloud Computing,” Nat. Inst. Standards Technol., Sep. 2011. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf
[9] A. K. Runchal, “The future of CFD and the CFD of the future,” *Comput. Thermal Sci., An Int. J.*, vol. 4, no. 6, pp. 517–524, 2012.
[10] C. Evangelinos and C. N. Hill, “Cloud computing for parallel scientific HPC applications: Feasibility of running coupled atmosphere-ocean climate models on Amazon EC2,” in *Proc. CCA-08*, Chicago, IL, Oct. 22–23,2008.
[11] R. Ledyayev and H. Richter, “High Performance Computing in a Cloud Using OpenStack,” in *5th Int. Conf. Cloud Comput., GRIDs, Virtualization*, Venice, Italy, May 25–29, 2014, pp. 108–113.
[12] P. OLeary, M. Christon, S. Jourdain, C. Harris, M. Berndt, and A. Bauer, “HPCCloud: A cloud/web-based simulation environment,” in *Proc. IEEE 7th Int. Conf. Cloud Comput. Technol. Sci.*, Vancouver, BC, Canada, 30 Nov–03 Dec. 2015, pp. 25–33.
[13] S. Guzzetti, T. Passerini, J. Slawinski, U. Villa, A. Veneziani, and V. Sunderam, “Platform and algorithm effects on computational fluid dynamics applications in life sciences,” *Future Gener. Comput. Syst.*, vol. 67, pp. 382–396, Feb, 2017.
[14] P. Kacsuk *et al.*, “WS-PGRADE/gUSE generic DCI gateway framework for a large variety of user communities,” *J. Grid Comput.*, vol. 10, no. 4, pp. 601–630, Dec. 2012.
[15] C. S. Liew, M. P. Atkinson, M. Galea, T. F. Ang, P. Martin, and J. I. Van Hemert, “Scientific workflows: Moving across paradigms,” *ACM Comput. Surv.*, vol. 49, no. 4, Dec. 2016.
[16] J. Wang, M. AbdelBaky, J. Diaz-Montes, S. Purawat, M. Parashar, and I. Altintas, “Kepler + CometCloud: Dynamic scientific workflow execution on federated cloud resources,” *Procedia Comput. Sci.*, vol. 80, pp. 700–711, Jun. 2016.
[17] G. Juve, M. Rynge, E. Deelman, J. S. Vockler, and G. B. Berriman, “Comparing futuregrid, Amazon EC2, and open science grid for scientific workflows,” *Comput. Sci. Eng.*, vol. 15, no. 4, 2013, Art. no. 6497031.
[18] Y. Zhao, L. Youfu, I. Raicu, L. Shiyong, W. Tian, and H. Liu, “Enabling scalable scientific workflow management in the Cloud,” *Future Gener. Comput. Syst.*, vol. 46, pp. 3–16, May 2015.
[19] T. Kiss, P. Kacsuk, E. Takacs, A. Szabo, P. Tihanyi, S. J. E. Taylor, “Commercial use of WS-PGRADE/gUSE,” in *Science Gateways for Distributed Computing Infrastructures*, P. Kacsuk, Ed. New York, NY, USA: Springer, 2014, pp. 271–286, doi: 10.1007/978-3-319-11268-8-19.
[20] CloudBroker, 2016. [Online]. Available: cloudbroker.com/documentation
[21] J. P. Braeunig, L. Brosset, F. Dias, and J. M. Ghidaglia, “Phenomenological study of liquid Impacts Through 2D Compressible Two-fluid Numerical Simulations,” in *Proc. 19th Int. Soc. Offshore Polar Eng. Conf.*, Osaka, Japan, Jul. 21–26, 2009.
[22] P. Hanafizadeh, M. H. Saidi, A. Nouri Gheimasi, and S. Ghanbarzadeh, “Experimental investigation of airwater, two-phase flow regimes in vertical mini pipe,” *Scientia Iranica*, vol. 18, no. 4, pp. 923–929, Aug. 2011.
[23] G. Terstyanszky *et al.*, “Validating scanned foot images and designing customized insoles on the cloud,” in *Proc. 49th Hawaii Int. Conf. Syst. Sci., IEEE*, 2016, pp. 3288–3296, doi: 10.1109/HICSS.2016.411.
[24] N. Grozev and R. Buyya, “Inter-Cloud architectures and application brokering: Taxonomy and survey,” *Softw. Practice. Experience*, vol. 44, no. 3, pp. 369–390, Mar. 2014.
[25] A. N. Toosi, R. N. Calheiros, and R. Buyya, “Interconnected cloud computing environments: Challenges, taxonomy, and survey,” *ACM Comput. Surveys*, vol. 47, no. 1, Jul. 2014, Art. no. 7.
[26] D. Bernstein, “Intercloud—The future cloud of clouds,” *J. Inst. Telecommun. Prof.*, vol. 7, no. 3, pp. 15–20, 2013.
[27] M. Cunha, N. C. Mendona, and A. Sampaio, “Cloud Crawler: A declarative performance evaluation environment for infrastructure-as-a-service clouds,” *Concurrency Computat., Pract. Exper.*, vol. 29, 2017, Art. no. e3825.
[28] O. Tomanek, P. Mulinka, and L. Kencl, “Multidimensional cloud latency monitoring and evaluation,” *Comput. Netw.*, vol. 107, part 1, pp. 104–120, Oct. 2016.

Simon J. E. Taylor received the B.Sc. (Hons) degree in industrial studies and the M.Sc. degree in computer studies both from Sheffield City Polytechnic, Sheffield, U.K., both in 1998, and the Ph.D. degree in distributed simulation from Leeds Metropolitan University, Leeds, U.K., in 1993.

He is a Professor of computing and the Director of the Modelling and Simulation Group with the Department of Computer Science, Brunel University London, Uxbridge, U.K.. He has a successful track record of research in modelling and simulation, particularly with high-performance distributed simulation in industry through digital infrastructures. He has developed these solutions with more than 20 companies resulting in over 5M GBP in cost savings and increased production. He has successfully applied these experiences in the development of digital infrastructures in Africa and has led innovation in more than ten African countries. He has also led the development of international standards in these areas. He regularly audits and chairs commercial project reviews in distributed computing and modelling and simulation. He has chaired several major conferences and has authored or coauthored more than 150 articles.

Dr. Taylor is a member of the ACM SIGSIM Steering Committee and Founder of the *Journal of Simulation.*

Anastasia Anagnostou received the B.Sc. degree in electronics engineering from Higher Technological Educational Institute of Thessaloniki, Greece, in 1999, the M.Sc. degree in telemedicine and e-Health systems from Brunel University London, U.K. in 2006, and the Ph.D. degree in hybrid distributed simulation from Brunel University London, in 2014.

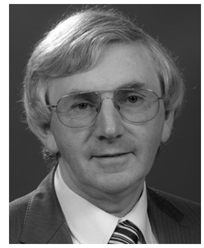
She was a Research Fellow with the Department of Computer Science, College of Engineering, Design and Physical Sciences, Brunel University London, Uxbridge, U.K. Her research interests include application of modeling and simulation techniques in the Healthcare and Industry.

Tamas Kiss received the M.Sc. degree in electrical engineering from the Faculty of Electrical Engineering, Budapest University of Technology and Economics, Hungary, in 1990, the M.Sc. degree in computer science and mathematics (with teaching qualification) from the Faculty of Natural Sciences, Eotvos Lorand University, Budapest, Hungary, in 2000, and the Ph.D. degree in distributed computing from the University of Westminster, U.K., in 2012.

He is a Reader in distributed computing with the Department of Computer Science, and the Director of the University Research Centre for Parallel Computing with the University of Westminster, London, U.K. He has attracted more than 20M GBP research funding and leads national and European research projects related to enterprise applications of cloud computing technologies. He was the Principal Investigator and a Project Coordinator of the CloudSME project. He currently leads the COLA H2020 project and an Application Experiments Coordinator in the CloudiFacturing H2020 project. He has authored or coauthored more than 100 papers in journals, conference proceedings, and as chapters of edited books. His research interests include distributed and parallel computing, cloud, cluster, and grid computing.

Gabor Terstyanszky received the M.Sc. degree in electronic engineering from St. Petersburg Electrotechnical University, St. Petersburg (previously Leningrad), Russia, in 1971 and the Ph.D. degree in computer science from the University of Miskolc, Hungary, in 1996.

He is the Director of the University Research Centre for Parallel Computing Centre at the University of Westminster, London, U.K. The Centre specializes in research on large-scale distributed computing infrastructures addressing research challenges of high-performance computing and high-throughput computing, such as clouds, clusters, and grids whereby large numbers of computing resources are used to process computationally demanding problems. He has been involved in more than 15 research projects as either Principal or Co-Investigator. He also had several research grants at universities in Germany, Spain, and U.K. He has published more than 140 papers in conferences and journals. He was a member of program committees of several conferences and workshops. He has supervised more than 10 Ph.D. students. He has taught several B.Sc. and M.Sc. modules on distributed computing and big data analytics. His research interests include distributed and parallel computing focusing on targeting research issues of Big Data and Cloud Computing.



Peter Kacsuk received the Kandidat degree in computer science (equivalent to Ph.D.) from the Hungarian Academy, Budapest, Hungary, in 1989, and the Doctor of Academy (D.Sc.) degree from the Hungarian Academy of Sciences, Budapest, Hungary, in 2001. He habilitated at the University of Vienna in 1997.

He is currently the Director of the Laboratory of Parallel and Distributed Systems at the Computer and Automation Research Institute of the Hungarian Academy of Sciences (MTA SZTAKI). He received the title of Professor from the Hungarian President in 1999, and he was a Full Professor with the University of Miskolc and with the Eötvös Lóránd University of Science Budapest. Since 2001, he has been a part-time Full Professor with the University of Westminster, London, U.K. He has authored or coauthored 2 books, 2 lecture notes, and more than 350 scientific papers on parallel computer architectures, parallel software engineering, grid and cloud computing.

Dr. Kacsuk is the Editor-in-Chief of the *Journal of Grid Computing* (Springer).

Nicola Fantini received the M.Sc. degree in computer science and business administration from the University of Zurich, Zurich, Switzerland, in 1991.

He is CEO of the CloudBroker GmbH, Zurich, Switzerland, and has been a Software Engineer, Project Manager, and entrepreneur in the field of software development for more than 20 years. He has been involved in and leading projects across industries, providing operational, commercial, financial, as well as technical solutions.

Djamel Lakehal received the M.Sc. degree in fluid mechanics from Ecole Centrale, France, in 1991, and the Ph.D. degree in the effect of surface roughness on turbulent flow past round-wall cylinders in 1994 from the University of Nantes, France.

He has until recently been the CEO of ASCOMP AG, Zurich, Switzerland. Upon the merge of ASCOMP with Pöyry, he was appointed as the Head of the Advanced Modeling and Simulation Section. In 1998, he joined the Institute of Energy Technology, ETH Zurich, as a Group Leader and a Senior Lecturer. He initiated the creation of the Computational Multifluid Dynamics Group. In 2004, he founded ASCOMP. He was hosted as a Invited Professor at MIT, KTH Stockholm, UC Santa Barbara, ENS Paris, and Imperial College London. He has authored or coauthored more than 100 journal papers and book chapters and monographs, and more than 120 conference papers in various areas related to fluid mechanics and heat transfer.

Dr. Lakehal acts as a Reviewer of research projects for the EU (DG Energy), and the U.S. Department of Energy.

Joris Costes received the Engineering degree in applied mathematics and physical modeling from ENSTA ParisTech, Paris, France, and the M.Sc. degree in applied mathematics in 2011 from ENS Paris-Saclay, Cachan, France.

He had the opportunity to pursue doctoral studies in numerical methods and scientific computing working with Prof. Jean-Michel Ghidaglia at ENS Paris-Saclay. He is currently an Aerospace Thermomechanical Engineer with the ArianeGroup near Bordeaux, France. He was an Engineer and a Project Leader with Eurobios, where he developed solutions in scientific computing for industrial companies. He moved to ArianeGroup in 2017 and is currently an Associate Researcher with ENS Paris-Saclay, CMLA laboratory.