Journal of Computational Science 25 (2018) 101–114

Contents lists available at ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs

VPH-HF: A software framework for the execution of complex subject-specific physiology modelling workflows

Marco Viceconti[a,*], Simone Bnà[b], Daniele Tartarini[a], Stelios Sfakianakis[c], James Grogan[d], Dawn Walker[e], Samuel Gamble[a], Debora Testi[b], CHIC Project Consortium[1]

[a] Department of Mechanical Engineering and INSIGNEO Institute for in silico Medicine, University of Sheffield, UK
[b] CINECA Supercomputing Centre, Italy
[c] Computational BioMedicine Laboratory, Institute of Computer Science, Foundation for Research and Technology, Greece
[d] Mathematical Institute, University of Oxford, UK
[e] Department of Computer Science and INSIGNEO Institute for in silico Medicine, University of Sheffield, UK

ARTICLE INFO

Article history:
Received 14 May 2017
Received in revised form 30 January 2018
Accepted 20 February 2018

Keywords:
In silico medicine
Workflow execution
Multiscale modelling

ABSTRACT

Computational medicine more and more requires complex orchestrations of multiple modelling & simulation codes, written in different programming languages and with different computational requirements, which when validated need to be run many times on large cohorts of patients. The aim of this paper is to present a new open source software, the VPH Hypermodelling Framework (VPH-HF). The VPH-HF overcomes the limitations of most workflow execution environments by supporting both Taverna and Muscle2; the addition of Muscle2 support makes possible the execution of very complex orchestrations that include strongly-coupled models. The overhead that the VPH-HF imposes in exchange for this is small, and tends to be flat regardless of the complexity and the computational cost of the hypermodel being executed. We recommend the use of the VPH-HF to orchestrate any hypermodel with an execution time of 200 s or higher, which would confine the VPH-HF overhead to less than 10%. The VPH-HF also provide an automatic caching system over the execution of every hypomodel, which may provide considerable speed-up when the orchestration is run repeatedly over large numbers of patients or within stochastic frameworks, and the input sets are properly binned. The caching system also makes it easy to form large input set/output set databases required to develop reduced-order models, and the framework offers the possibility to dynamically replace single models in the orchestration with reduced-order versions built from cached results, an essential feature when the orchestration of multiple models produces a combinatory explosion of the computational cost.

1. Introduction

The Virtual Physiological Human (VPH) is a framework of methods and technologies that enables the subject-specific modelling of complex physiological, pathological, and biological processes, across physiological systems and space-time scales [1]. VPH technologies make it possible to estimate quantities that are essential in supporting an accurate medical decision, but that are difficult or impossible to measure directly; for example, predicting the strength of a bone from CT data [2], or predicting the fractional flow reserve of a coronary stenosis form fluoroscopy images [3]. By collecting other, more easily accessible, quantitative information on the subject and combining it with the available mechanistic knowledge available for that specific disease process, VPH models can become useful tools to support the medical decision in a more personalised way [2,3].

VPH models are *integrative*, in the sense that they can capture and combine knowledge from multiple domains (biophysics, physiology, biology, pathology), multiple organ systems (cardiovascular and respiratory, neuromusculoskeletal, immune, endocrine, etc.), and across multiple space-time scales (typically from the molecular scale to the whole organism scale) [1].

While in some cases these multiple elements of knowledge are captured into a single, extremely complicated mathematical model, in most cases it is more convenient to capture each element into a separate model, which are then orchestrated into a single sim-

* Corresponding author at: Insigneo Institute for in silico Medicine, University of Sheffield, Pam Liversidge Building, Mappin Street, S13JD, Sheffield, UK.
E-mail address: m.viceconti@Sheffield.ac.uk (M. Viceconti).
[1] CHIC Project Consortium, Project FP7-ICT-600841, http://chic-vph.eu/consortium/.

https://doi.org/10.1016/j.jocs.2018.02.009
1877-7503/© 2018 Elsevier B.V. All rights reserved.

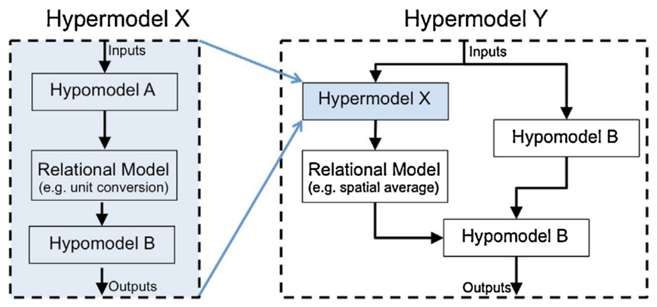

Fig. 1. Hypomodels and Relational models can be used to compose more complex models. On the left Hypomodels A and B are connected via a Relational model in the Hypermodel X. The process can be recursive, hypermodel X can be as hypomodel in hypermodel Y.

Table 1
Terminology mapping between VPH-HF, Taverna, and Muscle2.

VPH-HF	Muscle2	Taverna
Hypermodel	Model	Workflow
Hypomodel	Sub Model	Nested Workflow
Relational model	Conduit Filter (1:1 relationship), Mapper (N:M relationship) [4]	Shim, Beanshells [5]

ulation. Since there is not a single standardised terminology to refer to these models, hereinafter we will use the terminology used in the CHIC project: as such orchestrations are models of models, we will call them *hypermodels*. Consistently, each composing element will be referred to as *hypomodel*. Hypomodels can be further distinguished in *Component models*, which contain the modelling knowledge, and *Relational models*, which define how specific quantities predicted by one component model transform into the inputs required by another component model (Fig. 1).

Table 1 provides a mapping between the terminology used here, and the closest terms used for two other frameworks described below.

Several software tools are available to address this type of problem, including Discovery-Net [6], Kepler [7], VisTrails [8], Anduril [9], GridSpace2 [10], KNIME [11], Galaxy [12], Cuneiform [13], OpenAlea [14], Pegasus [15], and FabSim [16], just to name a few.

A systematic review of all these tools is beyond the scope of this paper; here we focus only on the two that are most relevant here. Taverna [5] and Muscle2 [4] are methodological and computational research frameworks to allow execution and coordination of multiple scientific software which implement computational models.

Taverna [5] uses a dataflow paradigm, well suited to address the needs of the bioinformatics community [17]. Bioinformatics models rarely require complex patterns of interaction and can be described with Directed Acyclic Graphs (DAG). Taverna also supports the integration of Web Services in the dataflow, since many bioinformatics databases are so large that they can only be queried remotely. More recently, support for conditional and cyclic data flows has been added, but its implementation is optimised for low numbers of iterations (for example, in loops workflows are executed serially, with their full instantiation overhead). Taverna provides an intuitive graphical user interface (Taverna Workbench) that, allowing to build scientific workflows with minimal computational expertise, has been adopted in different domains: bioinformatics [17,18], biodiversity and ecology [19], astronomy [19,20], heliophysics [21], statistics [18], and systems biology [22].

Muscle2 addresses the simulation needs of complex biophysics models, such as those used to describe mechano-biology processes. These are frequently modelled with multiscale, strongly coupled models, that pose considerable computational efficiency issues [23–25]. Muscle2 includes a domain-specific language called Multiscale Modelling Language (MML) that allows an elegant formalisation of multiscale problems where several sub-models are coupled [25,26]. The Muscle2 software suite allows an efficient distributed execution of the models [4,24,27]. Muscle2 has been used to model and simulate multiscale models in a range of different domains: Cerebrovascular blood flow [23], irrigation network [28], Reverse engineering of gene regulatory networks [29], turbolence transport [30], in-stent restenosis [31] and biomedical simulations [23]. Recent development on the Muscle2 has added support for an optimised MPI interface in C++ to better exploit parallel HPC systems [27,32].

The aim of this paper is to present a new open source software, the VPH Hypermodelling Framework (VPH-HF), which was developed in the frame of the EC-funded project “Computational Horizons In Cancer (CHIC): Developing Meta- and Hyper-Multiscale Models and Repositories for In Silico Oncology” (FP7-ICT-600841), hereinafter referred to as the CHIC project. The VPH-HF was developed in the attempt to address the needs of the modellers in the cancer research community: a rapid prototyping platform with reusable multiscale models that could interact using different graph topologies and capable of processing clinically relevant data. Our initial specifications analysis showed that some requirements would have been met by Taverna, some by Muscle2, and some by neither of these. Thus, rather than re-inventing the wheel, we designed VPH-HF on top of Taverna and Muscle2, integrating the two frameworks, and adding functionalities only where necessary. Modelling is an iterative multistage activity, where the model evolves through a gradual process from formulation through verification, validation, uncertainty quantification and ultimately optimisation. The CHIC project computational architecture offers a collection of services that make this iterative process less time-consuming, including a web based graphical editor, a model continuous integration system, a set of repositories for data, metadata and models, and a web-based results dashboard to present the model outputs to the clinical end-users. At the centre of this complex architecture, the VPH-HF ensures the required level of execution services.

2. The VPH hypermodelling framework

2.1. General specifications and paradigm

The CHIC project computational architecture is schematically represented in Fig. 2.

Users access the system through three interfaces: modellers design their models with a visual editor called the *VPH Hypermodelling Editor* (VPH-HE), and manage their execution directly from the VPH-HE or through the Administration & dashboard interface of the VPH-HF; clinical users request the execution of existing hypermodels on selected clinical data and see the results through a specialised web-based user interface (called *Clinical Research Application Framework*, CRAF) that queries the relevant repositories. Users are authenticated through a proprietary identity provider developed by Custodix,[2] which exposes a standard Security Assertion Mark-up Language 2.0 (SAML 2.0) interface; the VPH-HF source code can be easily modified to work with any SAML2-based identity providers. Similarly, the hypermodels, simulations, and clinical data repositories are proprietary implementations of the CHIC consortium, not available under an open source license. The interaction of the VPH-HF with the CHIC repositories is based on standard Web Service interfaces, so the VPH-HF source code can be easily mod-

[2] https://www.custodix.com.