Workflows for Wind Tunnel Grid Applications

A. Paventhan[1], Kenji Takeda[1], Simon J. Cox[1], and Denis A. Nicole[2]

[1] Microsoft Institute for High Performance Computing,
School of Engineering Sciences,
University of Southampton, UK
{povs, ktakeda, sjc}@soton.ac.uk
[2] School of Electronics and Computer Science,
University of Southampton, UK
dan@ecs.soton.ac.uk

Abstract. Aerodynamicists use wind tunnels to aid the research, design and development of products such as aircraft, cars, and yachts, amongst others. The data acquired from such tests must be acquired, collated, processed and analysed in a timely fashion to maximise productivity. In many scenarios a variety of data acquisition systems are used, and managing the overall testing process can be challenging. The wind tunnel grid project aims to provide an extensible, network-based system that can provide a more seamless working environment for scientists and engineers, so that they can focus on the data analysis and interpretation part of the process.

In this paper we describe the development and implementation of the wind tunnel grid system workflow. By exploiting Windows Workflow Foundation we are able to provide an easy-to-use and extensible workflow environment (wind tunnel grid workflow framework) that meets the requirements of both the developer and end-user well. By leveraging the .NET-based CoG Toolkit previously developed, interoperability with Globus grid services is demonstrated.

1 Introduction

The importance of workflows while developing real-world scientific and grid applications is becoming increasingly apparent [1]. While many business workflow systems tend to be control flow driven, many scientific workflows are data-centric.

Current Grid computing [2] solutions allow large-scale multi-institutional access to distributed resources. There are many issues, however, still to address while implementing application-specific scientific workflow on grids.

We can categorize scientific workflow systems as below: 1) Workflow systems motivated by application driven system requirements. 2) Generic workflow systems. The extensibility of a domain-specific workflow solutions to another application domain is limited as it does not address other sets of requirements. On the other hand, a generic workflow solution would require further customization to be applicable to a particular application domain. The set of requirements for scientific workflow systems in different scientific disciplines, in terms of data

V.N. Alexandrov et al. (Eds.): ICCS 2006, Part III, LNCS 3993, pp. 928–935, 2006.

size, formats, realtime requirements and computational complexities, bring different sets of challenges. Considering the Wind Tunnel user requirements (see Section 3), a customized workflow is important due to the varied nature of the work, which changes on a job-by-job basis in a multi-user facility .

In this paper we present an approach to building customized scientific workflows based on WinFX Windows Workflow Foundation. Windows Workflow Foundation is an extensible framework for developing workflow solutions with predefined set of activities (IfElse, While, Parallel, InvokeWebService and so on) and support for building domain-specific custom activities by inheritance. By making use our earlier work, the multi-language Commodity Grid Kit [3], we deliver a wind tunnel grid workflow framework for application development.

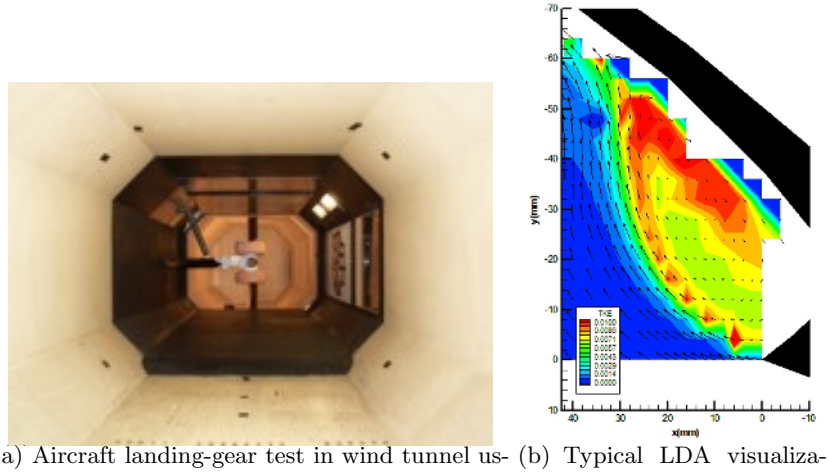
The rest of the paper is organized as follows. In Section 2 we discuss related works. Section 3 outlines wind tunnel experiment requirements to illustrate why workflow customization is important. Section 4 covers a brief overview of Windows Workflow Foundation. In Section 5, we present our approaches to wind tunnel grid workflow, with conclusions and future work presented in Section 6.

2 Related Works

In this section, we discuss the influences of related works on our project and highlight the particular wind tunnel experimental requirements that lead us to our work.

The GriPhyN [4] project addresses the workflow requirements of physics experiments. On data request, an abstract workflow is constructed. It is then converted to concrete workflow represented as Condor DAGman files [5] for submission to Condor-G scheduler. In case of wind tunnel experiments, the workflow is triggered by data acquisition and the raw data transfer requires more customization. In the KEPLER system [6], the workflow components are known as *actors* and controlled by an object known as a *director*. KEPLER addresses Grid and web services access, but, the integration of data acquisition hardware and experiment specific data transfer are paramount in our work. Grid-DB [7] is a data-centric grid workflow system. It provides a declarative language to define workflow. DiscoveryNet [8] addresses the need for knowledge discovery process in lifesciences. The components and workflows in DiscoveryNet are composed as Web and Grid services by sharing across teams. While the Triana [9] problem solving environment demonstrates distributed workflow to implement gravitational wave search algorithm.

The wind tunnel experiments run on several proprietary systems and their integration into scientific workflow requires customized solutions. Also, the data transfer and processing requirements are experiment specific, with the issues of physical locality before, during and after any given test a significant issue . The wind tunnel workflow framework is aimed at providing user with ready-to-use experiment specific workflow activities, hiding the underlying complexities and at the same time providing user with an option for customization, if required. Over time the processes and systems available in the wind tunnel change significantly, so the ability to rapidly develop and customise workflows is a necessary requirement.



(a) Aircraft landing-gear test in wind tunnel using microphone arrays

(b) Typical LDA visualization [11]

Fig. 1. Wind Tunnel Experiments

3 Wind Tunnel Experiments

The wind tunnel facilities at University of Southampton [10] are typical in that they house a variety of specialized experimental hardware and software for academic and industrial research. They are used in a wide variety of projects including fundamental aerodynamics, aerodynamics of racing cars and road vehicles, rotorcraft aerodynamics, aeroacoustics, aeronautics, wind engineering and industrial aerodynamics. Currently, in many wind tunnel experiments, the data movement operations to the processing computer are manual due to interoperability issues between hardware, software and the acquisition systems. The automated solution would require the following steps: 1. Experiment-specific data verification to ensure whether the acquisition was indeed successful 2. Experiment-specific annotation of metadata for auto-upload and processing 3. Raw data movement operations (based on metadata) and 4. User-defined processing step. We discuss below the requirements of a typical wind tunnel experiment that has been used to demonstrate the Grid system.

3.1 Laser Doppler Anemometry (LDA)

LDA systems use non-intrusive point-measurement techniques to accurately measure fluid velocity in highly turbulent or reversing flow. For each experimental configuration, a calibration step must be performed and a transformation matrix must be derived. LDA data acquisition software collects selected number of samples (typically thousands) at user programmed traverse positions of up to three velocity components (This value is also equal to number of Burst Spectrum Analysers (BSA)). The collected data are stored in separate raw data files for each traverse position. The raw data filenames have a suffix 0, 1 or 2 to indicate the velocity component (u,v & w, in the laser coordinate system). The file extension

represents the traverse position. There are essentially $n \times p$ raw data files for one experiment, n *(n=3)* is the number of velocity component and p is the number of traverse positions. The user parameters for acquisition are stored in a separate flat file. The upload activity of the workflow requires verification of the raw data files prior to the uploading to processing node. Since the number of velocity components and traverse positions are known from the parameter file, all the raw data files along with metadata (transformation matrix, user parameters) can be uploaded without any user intervention. The LDA processing steps include data conversion, coincidence processing, moment processing, spectrum processing and correlation processing. The user should be able to run the default algorithm or provide the new one by overriding the default.

The other experiments that we work on with similar requirements include: 1. The microphone phased array technique used in aeroacoustic research with near-realtime upload and processing requirements. and 2. Particle Image Velocimetry (PIV) with binary image upload and high-performance cross-correlation processing requirements. As can be seen from the requirements, the metadata must be used for customized data upload. In addition, the user processing step also requires customization so that different algorithms can be developed and used as the state-of-the-art advances.

4 Windows Workflow Foundation

Microsoft Windows Workflow Foundation is an extensible framework and is part of the upcoming Microsoft's next generation development Framework, WinFX [12]. The workflow in Windows Workflow Foundation is composed from a set of *activities*, compiled to a .NET assembly and is executed on the workflow runtime and the Common Language Runtime (CLR).

4.1 Workflow Model and Composition

There are two models supported [13]: 1. Sequential Workflow Model—comprising activities that execute in a predictable sequential path, and 2. State Machine Model—a flow driven by events triggering state transitions. In both these models the basic element of the workflow is called an *activity*. Some of the Windows Workflow Foundation's activity types include: control-flow (While, IfElse, Delay), exception (throw, exception-handler and BPEL compensations), data handling (Update, Select), transactions (and compensations for long-lived "transactions" that cannot be directly unwound) and Communication (InvokeWebService, InvokeMethod). A workflow consists of metadata for the workflow definition and the accompanying .NET classes that form the code file. The workflow can be composed using a visual workflow designer, which has a drag-and-drop interface or declaratively in XOML, an XML dialect for writing workflows. The workflow can also be completely coded in CLR languages. A workflow must be compiled with *wfc* workflow compiler before it can be run. All the Windows Workflow Foundation *activities* are derived from *Activity* base class. The Windows Workflow Foundation extensible development model enables creation of

domain-specific *activities* which can then be used to compose workflows that are useful and understandable by domain scientists.

4.2 Workflow Runtime, Scheduling and Hosting

The Workflow runtime layer is at the core of Windows Workflow Foundation and is responsible for execution, tracking, state management, scheduling and policies. The workflow engine runs inside a *hosting process* provided by the workflow application. The hosting layer is responsible for communication, persistence, tracking, transaction, timing and threading. It is possible to dynamically update the running workflows on the fly. With this flexible approach to workflow hosting and extensible framework for workflow activities, most of the functionality of the state-of-the-art scientific workflow systems [14] can be hosted on top of Windows Workflow Foundation. In the following sections we illustrate how specific workflows to wind tunnel aerodynamic testing can be constructed using Windows Workflow Foundation.

5 Wind Tunnel Grid Workflow

In this section, we describe architecture of the wind tunnel application workflow and show how customized solutions can benefit the wind tunnel users.

5.1 Application Workflow Leveraging Windows Workflow Foundation

Our approach to implementing a wind tunnel grid workflow based on Windows Workflow Foundation is shown in Figure.2. Users are able to use local services, Web Services and Globus Grid services using our previously developed MyCoG.NET commodity grid toolkit [3]. Three different sets of activities are available to compose an experimental workflow: 1. Windows Workflow Foundation Activities 2. MyCoG.NET-based Grid activities to access Globus services 3. Experiment-specific activities for upload, processing, results and so on. The user can design the workflow from these activity sets depending on his or her requirements.

There are two possible options to host the workflow: 1. Client-controlled hosting, and 2. By submitting to the wind tunnel grid workflow server for hosting. In client controlled hosting, the workflow runtime runs as part of the *host process* running on the user's PC. In this case, the user has to leave the *host process* running until the workflow finishes. Underlying the hosting process is the WinFX workflow runtime and .NET Common Language Runtime. The workflow can be monitored from wind tunnel grid workflow client while it is running. In the second case, the user deploys their workflow for hosting to the wind tunnel grid workflow server after successful Grid Security Infrastructure (GSI) [15] authentication and delegation of user credentials. The wind tunnel grid workflow server maintains user account information. A separate *host process* is instantiated for the user's workflow and the runtime is started. This allows the user to disconnect after submission and monitor the workflow periodically from wind tunnel grid workflow client.

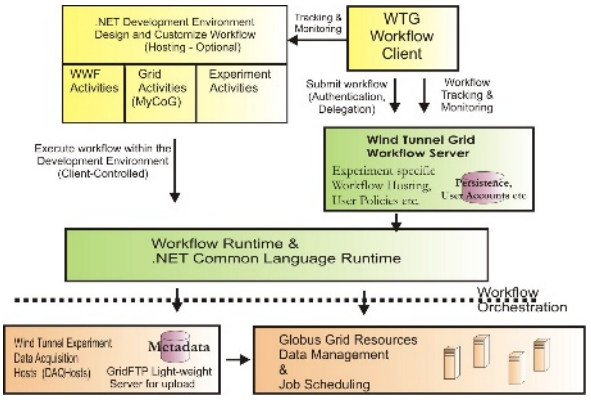


Fig. 2. Wind Tunnel Experimental Workflow Architecture

The generic wind tunnel workflow activities are: *WTWInit* - initializes wind tunnel workflow server hosting process for the user; this is the first activity in any wind tunnel experimental workflow. *UserNotification* - Customized user notification on the state of the workflow (workflow completion or failure). Figure.3 shows a sequential LDA workflow designed using customized wind tunnel grid workflow activities. The WaitForDAQ is an event driven activity customized for the LDA experiment. On completion of the data acquisition, this activity verifies the raw data files for completeness and workflow transitions to next activity. The MyGridFTP, MyGram and MyMDS activities use the MyCoG.NET Commodity Toolkit to access Globus resources. The implementation details of MyCoG.NET are discussed in [3]. These Grid service access activities are further customized for individual experiments. For example, as shown in Figure.3, the *LDAUpload* activity derived from MyGridFTP has specific input properties for the experiment (data acquisition hostname, number of data points, number of burst spectrum analysers). Some properties are initialized at workflow design time with default values and others received as input from the *host process*. The experiment specific properties would enable, for example, automatic uploading of raw data files from data acquisition host to a Gram server, as can be seen from Section.3.1. The *FetchResults* is an activity derived from MyGridFTP to transfer results from Gram host to Windows Workflow server and to the user's desktop.

The microphone array processing algorithm is significantly more processor intensive than the LDA case. A similar workflow can be constructed using Globus services, or Windows-based HPC resources. This is currently carried out using Labview and Matlab, but a C# version of the processing algorithms is being developed and integrated using Windows Workflow Foundation.

We specifically highlight the ease with which the Windows Workflow version of the LDA system was constructed. Due to the ease-of-use and intuitive nature of the workflow GUI, the user is able to drag-and-drop activities onto the work surface, rapidly customising their own workflow without worrying about lower-level details.

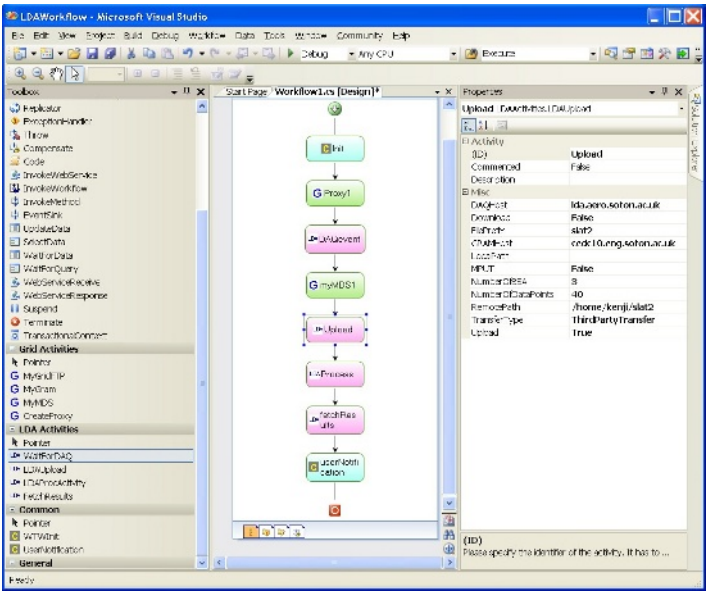


Fig. 3. LDA workflow

In our earlier implementation [3] of LDA workflow, each stage of the workflow is triggered by the user via portal interface. By bringing Globus grid service access to the Windows Workflow Foundation using MyCoG, the user would be able to design, execute and monitor wind tunnel experimental workflow on Globus resources.

This is of significant benefit to the wind tunnel engineer, who previously had to deal with the workflow steps on different hardware using bespoke software tools. The new system provides benefits not only in terms of speed, but reliability and consistency for the testing process and subsequent analysis.

6 Conclusions and Future Work

In this paper we have discussed the implementation of real-world scientific workflows using Windows Workflow Foundation in wind tunnel applications. This provides an easy-to-use, customisable and extensible framework to create sequential or state-based workflows. We have shown, using the MyCoG.NET commodity grid toolkit, that interoperability between Windows Workflow Foundation and Globus grid services is possible, including certificate-based authentication, proxy support, GridFTP file transfer and GRAM job submission. In this way Windows Workflow Foundation can be used to orchestrate workflows between Microsoft Windows-based Services, other Web Services and Globus Grid services.

While the paper describes an application-specific example of using Windows Workflow Foundation, it is designed as a generic framework. It is therefore suitable for other branches of experimental and computational sciences as well. Due

to its extensibility, the addition of provenance tracking, semantic definition and representation, and metadata derivation from results is possible.

As more grid resources move to Web Service interfaces, the use of Windows Workflow Foundation as a generic framework for composing scientific workflows is likely to become more common.

References

1. Ludascher, B., Goble, C.: Guest Editors' Introduction to the Special Section on Scientific Workflows. ACM SIGMOD Record **34**(3) (2005)
2. Foster, I., Kesselman, C.: The GRID 2: Blueprint for a New Computing Infrastructure. second edn. Morgan-Kaufmann (2003)
3. Paventhan, A., Takeda, K.: MyCoG.NET:Towards a multi-language CoG Toolkit. In 3rd ACM International Workshop on Middleware for Grid Computing. (2005)
4. Deelman, E., Blythe, J., Gil, Y., Kesselman, C.: "Workflow Management in GriPhyN" in *Grid Resource Management J. Nabrzyski,et.al,eds.* Kluwer (2003)
5. Condor DAGman. (http://www.cs.wisc.edu/condor/dagman/)
6. Ludscher, B., Altintas, I., et.al: Scientific workflow management and the KEPLER system. (Concurrency and Computation: Practice & Experience, (to appear))
7. Liu, D.T.: The design of GridDB: A Data-Centric Overlay for the Scientific Grid. In: Proceedings of the International Conference on Very Large Data Bases. (2004)
8. Curcin, V., Ghanem, M., et.al: IT Service Infrastructure for Integrative Systems Biology. In: IEEE International Conference on Services Computing. (2004)
9. Churches, D., Gombas, G., et.al: Programming scientific and distributed workflow with triana services. Concurrency and Computation: Practice & Experience, (to appear) (2005)
10. Southampton wind tunnels. (http://www.windtunnel.soton.ac.uk/)
11. Takeda, K., Ashcroft, G.B., Zhang, X., Nelson, P.A.: Unsteady aerodynamics of slat cove flow in a high-lift device configuration. AIAA Paper 2001-0706 (2001)
12. WinFX Developer Center. (http://msdn.microsoft.com/winfx/)
13. Andrew, P., Conard, J., et.al: Presenting Windows Workflow Foundation, Beta Edition. Sams (2005)
14. Yu, J., Buyya, R.: A taxonomy of scientific workflow systems for grid computing. ACM SIGMOD Record **34**(3) (2005) 44–49
15. Welch, V.: Grid Security Infrastructure Message Specification. (2004)