Accepted Manuscript

Software architectures to integrate workflow engines in science gateways

Tristan Glatard, Marc-Étienne Rousseau, Sorina Camarasu-Pop, Reza Adalat, Natacha Beck, Samir Das, Rafael Ferreira da Silva, Najmeh Khalili-Mahani, Vladimir Korkhov, Pierre-Olivier Quirion, Pierre Rioux, Sílvia D. Olabarriaga, Pierre Bellec, Alan C. Evans

PII: S0167-739X(17)30024-9
DOI: http://dx.doi.org/10.1016/j.future.2017.01.005
Reference: FUTURE 3287

To appear in: *Future Generation Computer Systems*

Received date: 16 May 2016
Revised date: 17 October 2016
Accepted date: 7 January 2017

Please cite this article as: T. Glatard, M.-ɾ Rousseau, S. Camarasu-Pop, R. Adalat, N. Beck, S. Das, R.F. da Silva, N. Khalili-Mahani, V. Korkhov, P.-O. Quirion, P. Rioux, S.D. Olabarriaga, P. Bellec, A.C. Evans, Software architectures to integrate workflow engines in science gateways, *Future Generation Computer Systems* (2017), http://dx.doi.org/10.1016/j.future.2017.01.005

Software architectures to integrate workflow engines in science gateways

Tristan Glatard[a,b,c], Marc-Étienne Rousseau[b], Sorina Camarasu-Pop[c], Reza Adalat[b], Natacha Beck[b], Samir Das[b], Rafael Ferreira da Silva[e], Najmeh Khalili-Mahani[b], Vladimir Korkhov[g], Pierre-Olivier Quirion[d], Pierre Rioux[b], Sílvia D. Olabarriaga[f], Pierre Bellec[d], Alan C. Evans[b]

[a]*Department of Computer Science and Software Engineering, Concordia University, Montreal, Canada.*
[b]*McGill Centre for Integrative Neuroscience, Montreal Neurological Institute, McGill University, Canada.*
[c]*University of Lyon, CNRS, INSERM, CREATIS, Villeurbanne, France.*
[d]*Centre de Recherche de l'Institut de Gériatrie de Montréal CRIUGM, Montréal, QC, Canada.*
[e]*University of Southern California, Information Sciences Institute, Marina del Rey, CA, USA.*
[f]*Dept. Clinical Epidemiology, Biostatistics and Bioinformatics, Academic Medical Center, University of Amsterdam, NL.*
[g]*St. Petersburg State University, Russia.*

Abstract

Science gateways often rely on workflow engines to execute applications on distributed infrastructures. We investigate six software architectures commonly used to integrate workflow engines into science gateways. In *tight integration*, the workflow engine shares software components with the science gateway. In *service invocation*, the engine is isolated and invoked through a specific software interface. In *task encapsulation*, the engine is wrapped as a computing task executed on the infrastructure. In the *pool model*, the engine is bundled in an agent that connects to a central pool to fetch and execute workflows. In *nested workflows*, the engine is integrated as a child process of another engine. In *workflow conversion*, the engine is integrated through workflow language conversion. We describe and evaluate these architectures with metrics for assessment of integration complexity, robustness, extensibility, scalability and functionality. Tight integration and task encapsulation are the easiest to integrate and the most robust. Extensibility is equivalent in most architectures. The pool model is the most scalable one and meta-workflows are only available in nested workflows and workflow conversion. These results provide insights for science gateway architects and developers.

Keywords: Workflow engines, science gateways, software architectures.

1. Introduction

Several software architectures can be adopted to integrate workflow engines in the ecosystem of tools and services offered by science gateways, with important consequences for the development effort required and resulting system.

This paper describes, illustrates and compares such architectures, based on system-independent representations of their main components and interactions. It is informed by our experience in the development and sustained operation of the CBRAIN [40], NSG [37, 38] and VIP [17] science gateways during the past 7 years, as well as by lessons learned from several science gateway and workflow projects such as SHIWA[1] and ER-flow[2].

This analysis is intended for experts of science gateway and workflow engine design. It is an abstraction effort to identify and evaluate the fundamental architectural patterns that are encountered while integrating workflow engines and science gateways. In real systems, such patterns sometimes coexist due to the historical and technical context of software projects.

The remainder of this section provides background information and definitions of workflow engines, science gateways and infrastructures. In Section 2, we describe six architectures within a consistent framework that underlines the functional in-

[1] `http://www.shiwa-workflow.eu`
[2] `http://www.erflow.eu`