# Multi-infrastructure workflow execution for medical simulation in the Virtual Imaging Platform

Rafael Ferreira da Silva, Sorina Camarasu-Pop, Baptiste Grenier, Vanessa Hamar, David Manset, Johan Montagnat, Jérôme Revillard, Javier Rojas Balderrama, Andreï Tsaregorodtsev, Tristan Glatard

# Multi-infrastructure workflow execution for medical simulation in the Virtual Imaging Platform

Rafael FERREIRA DA SILVA[1], Sorina CAMARASU-POP[1], Baptiste GRENIER[3], Vanessa HAMAR[2], David MANSET[3], Johan MONTAGNAT[4], Jérôme REVILLARD[3], Javier ROJAS BALDERRAMA[4], Andrei TSAREGORODTSEV[2], Tristan GLATARD[1]

[1] *Université de Lyon, CNRS, INSERM, CREATIS, 69621 Villeurbanne, FRANCE*
[2] *Centre de Physique des Particules de Marseille, 13288 Marseille, FRANCE*
[3] *maatG France, 74160 Archamps, FRANCE*
[4] *CNRS / UNS, I3S lab, MODALIS team*

**Abstract.** This paper presents the architecture of the Virtual Imaging Platform supporting the execution of medical image simulation workflows on multiple computing infrastructures. The system relies on the MOTEUR engine for workflow execution and on the DIRAC pilot-job system for workload management. The jGASW code wrapper is extended to describe applications running on multiple infrastructures and a DIRAC cluster agent that can securely involve personal cluster resources with no administrator intervention is proposed. Grid data management is complemented with local storage used as a failover in case of file transfer errors. Between November 2010 and April 2011 the platform was used by 10 users to run 484 workflow instances representing 10.8 CPU years. Tests show that a small personal cluster can significantly contribute to a simulation running on EGI and that the improved data manager can decrease the job failure rate from 7.7% to 1.5%.

**Keywords.** Workflows, EGI, cluster, pilot jobs, data management, web portal

## 1. Introduction

Medical imaging research increasingly relies on simulation for studying image analysis, treatment planing or instrumentation prototyping and simulation codes generally need specific computing infrastructures to produce realistic results.

In the last decade, the European Grid Infrastructure (EGI)[1] demonstrated its ability to support heavy scientific computations with a high throughput [**?**] and tools were developed to bring it to researchers. Such environments enabled large-scale computing campaigns in various fields including life-sciences. However, high computing throughput is not the only desired characteristic in life sciences. In particular, high latencies and low reliability hamper applications up to a point that prevented grids from being adopted as a daily scientific instrument in this community. Instead, when grid infrastructures are

[1] `http://www.egi.eu`

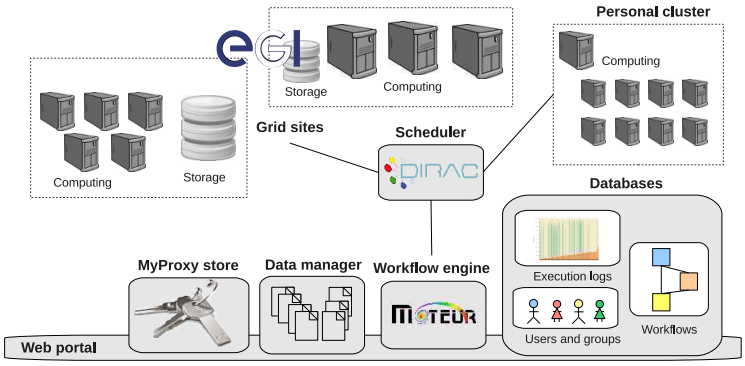


**Figure 1.** Architecture of the Virtual Imaging Platform.

used, they are combined with other computing facilities such as local clusters, multi-core CPUs or general-purpose GPUs. Multi-infrastructure personal computing systems are *de-facto* assembled, often manually.

Similar conclusions apply to data management. While grid can store and transfer important volumes of data with a high-throughput, it is not suitable for applications manipulating collections of small files that are sensitive to transfer errors and to latency. In fact, many experiments designated data transfers as an important cause of job failures [1].

The Virtual Imaging Platform (VIP) provides online access to imaging simulators. It describes pipelines as workflows made of jGASW activities that are executed with the MOTEUR engine using DIRAC pilot jobs. The overall architecture of the platform is diagrammed on Fig. 1. For now, core simulation workflows are available for medical image simulators of Magnetic Resonance Imaging (MRI), Ultrasound imaging (US), Computed Tomography (CT) and Positron Emission Tomography (PET) [2].

Core simulation is not the only process involved in a simulation. The platform will soon be extended with workflows and interfaces facilitating the handling of biological object models, simulator parameters and post-processing steps. These workflows only have limited parallelism and consist of jobs with various CPU/data ratios. Therefore, they are very sensitive to job and data transfer errors as well as to long waiting times.

Our goal is to set up a multi-infrastructure workflow execution platform supporting applications with mixed performance requirements. In this paper we describe how local personal clusters that are not part of a grid are integrated in the platform and how grid data management can be complemented by local storage to improve data transfer reliability.

Section 2 describes a jGASW extension to support the annotation of legacy simulation codes for execution on multiple infrastructures. Then Section 3 describes a DIRAC agent targeting non-grid clusters. Section 4 focuses on data management ; it explains how we use local storage to complement grid Storage Elements. The VIP web portal is described in Section 5 and usage results are presented in Section 6. The paper closes on a summary and related work.

## 2. Workflow wrapping of simulation codes for multi-infrastructure execution

The motivation for using workflows is twofold. At design time, it provides an accessible, high-level formal representation used to determine sub-processes that can be enacted

```
<implementation>
  <release>0.1.2</release>
  <platforms>
    <platform infrastructure="egee">
      <profiles>
        <profile job="normal">
          <target>ExecutionField-EGEE.sh</target>
          <boundArtifact>lfn://.../ExecutionField-EGEE.tar.gz</boundArtifact>
        </profile>
      </profiles>
    </platform>
    <platform infrastructure="pbs">
      <profiles>
        <profile job="normal">
          <target>ExecutionField-PBS.sh</target>
          <boundArtifact>lfn://.../ExecutionField-PBS.tar.gz</boundArtifact>
        </profile>
      </profiles>
    </platform>
  </platforms>
</implementation>
```

**Figure 2.** Multi-infrastructure code descriptor of a simulator component.

concurrently on distributed computing platforms. At runtime, the workflow is interpreted and executed using the MOTEUR workflow engine [3], which provides an asynchronous grid-aware enactor. Workflows are described using GWENDIA, a language dedicated to data-intensive scientific applications description. The VIP simulation workflows are extensively described in [2]. They consist in 3 sub-workflows that are object preparation, core simulation computation and post-processing.

The MOTEUR enactor is architected as a two-level engine. At the upper level, the core engine interprets the workflow representation, evaluates the resulting data flows and produces computational tasks ready for remote execution through a generic interface. At the lower level, the tasks descriptions are converted into jobs targeting a specific computing infrastructure. The interface between the core engine and the job generation is implemented by a component named jGASW [5].

jGASW provides a mechanism to package an application and its dependencies, to deploy it and to publish it. Once the application is packaged, jGASW handles file transfers and job (re-)submission to the computing platform. jGASW is composed of three independent but complementary modules: (i) a code wrapper bundling codes for execution on distributed platforms, (ii) a dynamic and extensible code invocation interface to adapt to different execution infrastructures, and (iii) a client-side invocation library. In jGASW the wrapping of applications relies on a code descriptor which formalizes the invocation interface of the command line and defines its dependencies. This code descriptor is created through a graphical user interface. A portable bundle containing the application and all its dependencies is generated after the description is filled.

When an application wrapped with jGASW receives data, it generates jobs in the form of bash scripts wrapped in grid jobs and submitted to the DIRAC system. Bash scripts first test file upload, then download the input data, launch the application command-line and finally upload the results.

To support multiple infrastructures, the jGASW code descriptor was modified to distinguish the application description from its implementation. The former contains the application meta-data (name, description, copyright) and descriptions for inputs and outputs. Several implementations can be defined. Each of them has a release number and is defined for one or several platforms (e.g. EGI or cluster). A platform contains execution profiles defining the job type (normal or MPI flavors). A profile holds information about

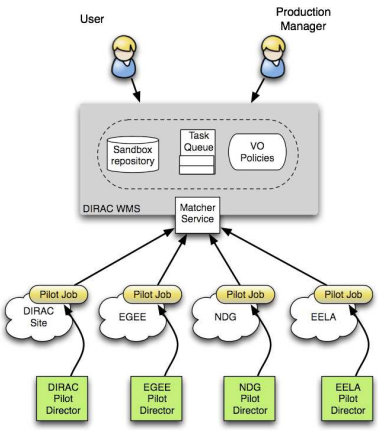


**Figure 3.** DIRAC Workload Management with pilot jobs.

the application bundle and target. The bundle contains all the application dependencies while the target is the main executable. In addition, customized execution environments can be defined for a specific profile or shared among platforms and releases. Fig. 3 shows a descriptor sample for a simulator implementation defined on two infrastructures.

## 3. Multi-infrastructure execution with DIRAC

The DIRAC [6] Workload Management System (WMS) is represented on Fig. 4. It implements a "pull" scheduling in which all the jobs are submitted to the central Task Queue and pilot jobs are sent to Worker Nodes at the same time. Pilot jobs run special agents that "pull" user jobs from the Task Queue, set up their environment and steer their execution. Advantages of this approach are summarized below.

First of all, pilot jobs check their execution environment before retrieving user jobs, which reduces the failure rate of user jobs and increases the efficiency of the WMS. Besides, pilot jobs effectively reserve their Worker Node for the time slot allocated by the batch system, which reduces considerably the load on the grid middleware. In addition, the centralization of user jobs in the Task Queue enables community policy enforcements while decentralized priority rules defined at the site level are imprecise and suffer scalability problems. Finally, pilot jobs can easily incorporate computing resources of different nature and belonging to various administrative domains. Indeed, once running, pilot jobs behave the same regardless of the type of resources.

In VIP, DIRAC was extended to support non-grid clusters so that both EGI sites and local personal clusters are available to execute jGASW jobs. A few design constraints were identified for this cluster extension. First, no intervention from the cluster administrator must be required, i.e. everything should happen in the user space. In particular, installing a gLite computing element is out of question. Second, security rules must be respected. In particular, clusters must be used with distinct personal user accounts (no sharing of generic user accounts) and login/passwords cannot be stored in the VIP or in DIRAC. Finally, the solution should rely on minimal technical assumptions on the cluster architecture, in particular concerning network connectivity and software installations.

The proposed solution extends DIRAC with a cluster agent launched by the user on his cluster(s) account(s). The user logs in using his/her usual authentication method, independently from the DIRAC system. Once set up, the cluster agent runs a daemon that contacts the DIRAC Workload Management Service periodically. If there are waiting tasks for the current user, it launches DIRAC pilots to the local cluster queue. Once the pilots are running, they behave exactly as described in the previous section, but they can only retrieve tasks belonging to the user on behalf of whom they are running. As for any other computing site, the scheduling between local and remote cluster is done by DIRAC.

The cluster agent is released as a software bundle containing the user X.509 certificate, the VLET[2] grid data transfer client and DIRAC installation scripts. During the setup phase, the daemon is adjusted to the scheduler type (PBS, BQS and SLURM are supported). The user proxy is first created by the agent in the setup phase and then periodically renewed from a MyProxy server[3]. It is used for data transfers and to authenticate to the DIRAC server. A cron job is set to automatically restart the daemon in case the cluster is rebooted. The only required software dependency is a Python interpreter.

## 4. Reliable data management

VIP relies on the EGI three-tier data management. At the lowest level, files are stored at different sites on disk or on tape using Storage Elements (SE) such as DPM, dCache, STORM or Castor. All these SEs expose a homogeneous SRM interface through which files can be listed and transfers can be requested. SRM locations are registered in a Logical File Catalog (LFC) that provides a single indexing space for distributed files. File replication is enabled by linking several SRM locations to the same logical name. This architecture can handle replication but it does not provide any mechanism to guarantee file availability. With availability statistics ranging from 80% to 95%, it is critical for applications to set up their own data management system.

In VIP, critical input files such as jGASW bundles are replicated daily on a few SEs to reduce their sensitivity to SE unavailability. Files are also cached by pilot jobs so that tasks executed by the same pilot do not transfer the same file multiple times. To speed-up transfers and reduce the load on the SEs, output files are stored on the site where they are produced; in case the computing site has no SE or it cannot be accessed, files are transferred to a central SE. These measures are useful but not enough. As shown in [1], the job error rate coming from data transfers is still in the order of 5% to 10%.

The solution proposed here is to complement grid SEs with a local data manager included the VIP. The data manager is used as a failover storage for files involved in a simulation. It can be accessed both by users and by grid jobs. The use-case is summarized on Fig. 5. When a simulation starts, MOTEUR replicates the input files on the data manager which is assumed reliable. These local copies are then used as failovers in case input files cannot be transferred. Jobs can also use for output transfers when their local SE is not accessible. Periodically, the data manager applies the following replication policy: (i) files are replicated to ensure a minimal number of replicas, (ii) above a certain threshold, replicas are deleted, (iii) files that are properly replicated are removed from the

[2] http://www.nikhef.nl/~ptdeboer/vlet

[3] http://grid.ncsa.illinois.edu/myproxy/

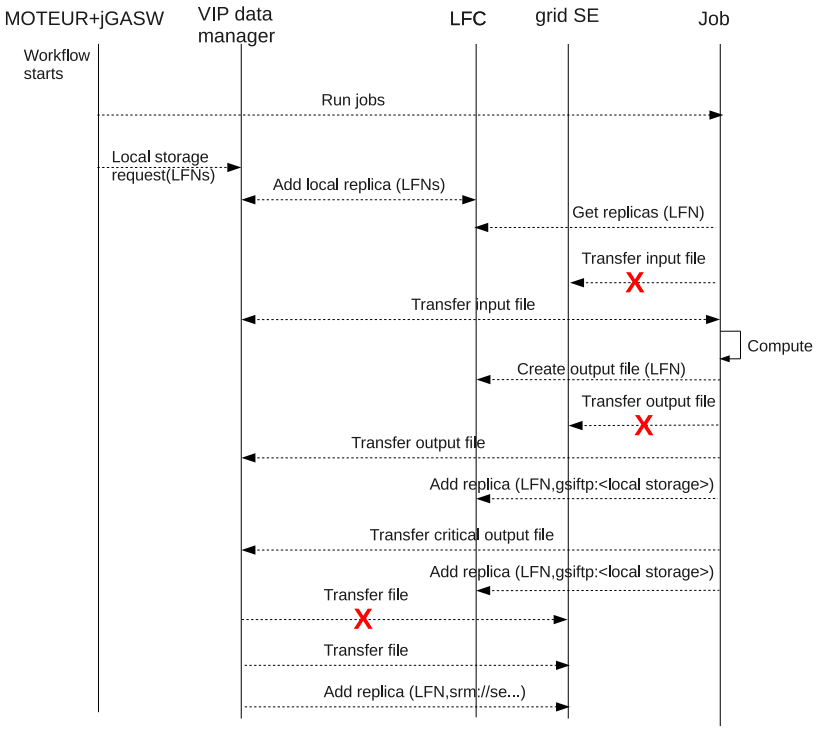


**Figure 4.** Local storage as a complement to grid SEs. Red crosses indicate transfer errors.

local DPM to ensure that the local storage is always available for critical file transfers, (iv) files that are not referenced in the LFC are deleted from the local storage.

The VIP data manager was implemented as an overlay of the DPM SE. It is seen as a regular SE except that it is not published in the information system. Therefore, it can be used with existing grid data clients and files stored in the local storage can be registered in the LFC as any other grid files[4].

## 5. Web Portal

The VIP portal [5] implements a client-server architecture based on servlets. It allows users to submit, monitor and manage their workflows. The portal also provides detailed performance information such as job flow diagrams and summaries of execution, upload and download times. Users are authenticated by X.509 certificates imported into their web browsers. To simplify grid authentication, an applet is available to create proxies directly from the browser keystore and upload them to a MyProxy server. All the subsequent grid operations are done with the user proxy.

Applications are organized in classes that users can access based on their group membership. When the user selects an application, the portal generates a form from its

[4] Checks in the information systems can be disabled in regular data transfer commands (`-nobdii` option).

[5] `http://vip.creatis.insa-lyon.fr`

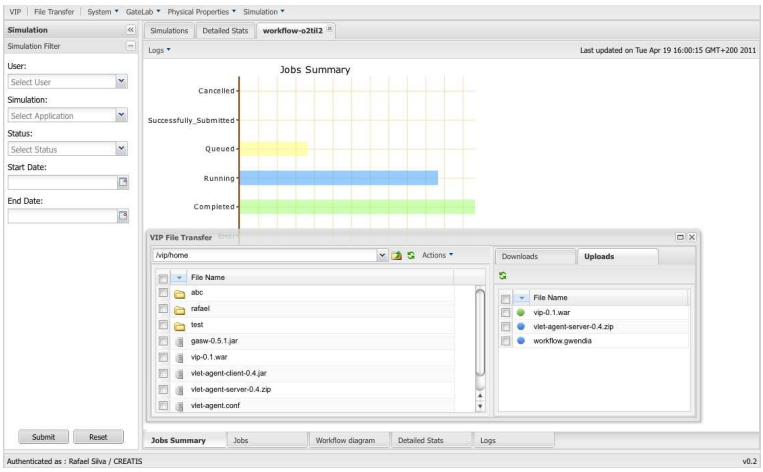


**Figure 5.** Screenshot of the VIP portal displaying the on-line monitor and file transfer tool.

workflow description. This form is filled in with input files and parameters and the workflow is submitted to MOTEUR. The portal provides statistics of workflow executions, jobs statuses and execution logs. Active workflows can be killed and inactive ones can be cleaned-up and purged.

Users can transfer files between their local machine and grid storage by using a file transfer tool. File upload is split in two steps: (i) upload from the local machine to the web portal server and (ii) file transfer to the grid through an asynchronous pool of transfers which is processed sequentially using the VLET library. Similarly, the download process consists in (i) transferring the file from the grid to the portal's server machine using the transfer pool and (ii) downloading the file to the user host. Fig. 6 is a screenshot of the portal displaying the on-line monitoring of a workflow and the file transfer tool.

## 6. Results

The portal using MOTEUR and DIRAC was put in production in November 2010. Since then, 484 workflow runs were performed by 10 users. Although it is primarily aimed for medical image simulation, it was used by various other applications including parameter studies in cardiac segmentation, Mean-Shift filtering and MR liver cartography as well as cancer treatment simulations with GATE [**?**]. Table 7 summarizes usage statistics for the main applications (successful jobs only).

The DIRAC cluster agent was tested on a PET simulation workflow described in [2]. It was deployed on a 134-node cluster with the maximal number of concurrently running pilots set to 67. Three workflow runs were conducted. Table 1 shows the job repartition between the local cluster and the EGI. This proof-of-concept test shows that involving a small personal cluster in a simulation running on EGI has a significant impact.

An experiment was conducted with an ultrasonic simulation to evaluate the impact of the data manager on a simulation running on EGI. Two runs of this simulation were considered. The data manager was enabled in the first one and it was disabled in the

| Application | Number of workflows | Total CPU time consumed |
|---|---|---|
| GATE simulation | 227 | 6.6 years |
| Liver cartography | 72 | 1.9 years |
| Mean-Shift filtering | 43 | 1.5 years |
| Image simulation | 40 | 198days |
| Cardiac Segmentation | 102 | 106 days |
| Total | 484 | 10.8 years |

**Table 1.** VIP usage (Nov 2010 - Apr 2011)

| Worfklow | #jobs EGI | #jobs local cluster |
|---|---|---|
| Run 1 | 101 | 77 |
| Run 2 | 129 | 37 |
| Run 3 | 148 | 16 |

**Table 2.** Multi-infrastructure execution of a PET workflow

| Data manager | Total submitted jobs | Data transfer errors | Total failed jobs |
|---|---|---|---|
| ON | 130 | 0 | 2 |
| OFF | 142 | 11 | 14 |

**Table 3.** Impact of the data manager on job reliability.

second one. The simulation consisted of 128 jobs dominated by data transfers. Each job downloaded 5 input files and uploaded 1 output file. Files were not replicated and an artificial failure rate of 1% was set on the data transfers. It correspond to the realistic case of an execution on the EGI where files are replicated twice[6]. Table **??** shows the impact of the data manager on job reliability. It displays the total number of submitted jobs, the number of data transfer errors and the total number of failed jobs for both runs.

As expected, the data manager zeroed the number of data transfer errors. In this case the remaining 2 failures are due to application errors on the computing nodes. The impact of data transfer errors on the application is important: when the data manager is not used, the observed job failure rate due to data transfer errors is 7.7% (11 failures for 142 submitted jobs). It comes from the fact that the 1% failure probabilities accumulate for all the 6 files transferred by the jobs.

## 7. Conclusions

The Virtual Imaging Platform can execute workflows on multiple infrastructures by coupling the MOTEUR engine with the DIRAC pilot-job system. The jGASW application descriptor was extended to support the description of applications on multiple infrastructures. Thus, application releases and environment can be adapted to each platform. DIRAC was extended to support personal clusters with no administrator intervention and respecting common security rules. Users can exploit different resources to which they

[6]Reported availability/reliability values on EGI are in the order of 90%. If files are replicated twice then the transfer failure rate is 1%.

have access in a uniform way by simply launching the cluster agent on each of them. This system was successfully tested on a PET simulation workflow. Results show that even a small personal cluster can significantly contribute to a simulation running on EGI.

A data manager was presented to complement grid Storage Elements with local reliable storage. The data manager is implemented as an overlay to the DPM Storage Element ; therefore it can be used with the Logical File Catalog and can be accessed using regular grid clients. It holds files involved in the active workflows and it is used as a failover in case grid storage is not available. It includes a daemon that periodically replicate the oldest files to the grid to ensure that incoming transfer requests can always be fulfilled. This data manager will be put in production in the coming months. A first test shows that it can decrease the job failure rate from 7.7% to 1.5%.

A web portal was developed to access the VIP. Workflows are organized in application classes that are available based on user group membership. It also has a file transfer tool so that workflows can be launched from a web browser only. Between November 2010 and April 2011, VIP was used by 10 users to run 484 workflow instances representing 10.8 CPU years. In the following months this portal will be extended with interfaces specific to medical image simulation.

## 8. Related work

This development started from the Virtual Laboratory for Medical Imaging [7] which uses MOTEUR to execute workflows on EGI with the DIANE pilot-job framework [**?**] and transfers input and output files with the VBrowser client application. The main motivations for the VIP development were (i) client configuration issues with the VBrowser and (ii) the adoption of a pilot-job system supporting user communities.

Along with jGASW, tools such as the LONI Pipeline [8], GEMLCA [9], gRAVI [10] or Soaplab2 [11] provide wrapping frameworks for applications.

Castor, dCache, DPM and STORM are the existing SRM implementations. To the best of our knowledge, none of them include a failover local storage as described here. We chose to conduct our developments in DPM to facilitate the integration with the existing EGI data management system.

Most of the large-scale experiments conducted on the EGI now use a pilot job system. Frameworks include DIANE [**?**], DIRAC [6], Condor glideIns [**?**] and WISDOM [**?**]. DIRAC was chosen for several reasons including fault tolerance, security, MPI support, compatibility with gLite and development sustainability.

## Acknowledgment

This work is funded by the French National Agency for Research under grant ANR-09-COSI-03 “VIP”. V. Hamar is supported by the GISELA Project under grant agreement no: 261487. We are grateful to Fabrice Bellet for his daily assistance with the Creatis cluster and IT setup. We thank the European Grid Initiative for coordinating the European-wide grid infrastructure, including technical support. We also thank our colleagues at France-Grilles for the operational support and technical discussions.

## References


[1] T. Li, S. Camarasu-Pop, T. Glatard, T. Grenier, and H. Benoit-Cattin. Optimization of mean-shift scale parameters on the egee grid. In *Studies in health technology and informatics, Proceedings of Healthgrid 2010*, volume 159, pages 203–214, 2010.

[2] A. Marion, G. Forestier, H. Benoit-Cattin, S. Camarasu-Pop, P. Clarysse, R. Ferreira da Silva, B. Gibaud, T. Glatard, P. Hugonnard, C. Lartizien, H. Liebgott, J. Tabary, S. Valette, and D. Friboulet. Multi-modality medical image simulation of biological models with the Virtual Imaging Platform (VIP). In *IEEE CBMS 2011*, Bristol, UK, 2011. submitted.

[3] Tristan Glatard, Johan Montagnat, Diane Lingrand, and Xavier Pennec. Flexible and efficient workflow deployement of data-intensive applications on grids with MOTEUR. *International Journal of High Performance Computing Applications (IJHPCA) IF=1.109 Special issue on Special Issue on Workflows Systems in Grid Environments*, 22(3):347–360, August 2008.

[4] Johan Montagnat, Benjamin Isnard, Tristan Glatard, Ketan Maheshwari, and Mireille Blay-Fornarino. A data-driven workflow language for grids based on array programming principles. In *Workshop on Workflows in Support of Large-Scale Science(WORKS'09)*, , pages 1–10, Portland, USA, November 2009. ACM.

[5] Javier Rojas Balderrama, Johan Montagnat, and Diane Lingrand. jGASW: A Service-Oriented Framework Supporting High Throughput Computing and Non-functional Concerns. In *IEEE International Conference on Web Services*, ICWS 2010, Miami (FL), USA, July 2010. IEEE Computer Society.

[6] A Tsaregorodtsev, N Brook, A Casajus Ramo, Ph Charpentier, J Closier, G Cowan, R Graciani Diaz, E Lanciotti, Z Mathe, R Nandakumar, S Paterson, V Romanovsky, R Santinelli, M Sapunov, A C Smith, M Seco Miguelez, and A Zhelezov. DIRAC3 . The New Generation of the LHCb Grid Software. *Journal of Physics: Conference Series*, 219(6):062029, 2009.

[7] Silvia Olabarriaga, T. Glatard, and P.T. de Boer. A virtual laboratory for medical image analysis. *IEEE T Inf Technol B*, 14(4):979–985, 2010.

[8] Ivo D. Dinov, John D. Van Horn, Kamen M. Lozev, Rico Magsipoc, Petros Petrosyan, Zhizhong Liu, Allan MacKenzie-Graham, Paul Eggert, Parker Douglas S., and Arthur W. Toga. Efficient, Distributed and Interactive Neuroimaging Data Analysis Using the LONI Pipeline. *Frontiers in Neuroinformatics*, 3(22):1–10, 2009.

[9] Thierry Delaitre, Tamas Kiss, Ariel Goyeneche, Gabor Terstyanszky, Stephen Winter, and Peter Kacsuk. GEMLCA: Running Legacy Code Applications as Grid services. *Journal of Grid Computing*, 3(1):75–90, 2005.

[10] Kyle Chard, Wei Tan, Joshua Boverhof, Ravi Madduri, and Ian Foster. Wrap Scientific Applications as WSRF Grid Services Using gRAVI. In *International Conference on Web Services*, ICWS'09, Los Angeles (CA), USA, July 2009.

[11] Martin Senger, Peter Rice, Alan Bleasby, Tom Oinn, and Mahmut Uludag. Soaplab2: More Reliable Sesame Door to Bioinformatics Programs. In *Bioinformatics Open Source Conference*, BOSC'08, Toronto, ON Canada, July 2008.