Medical image registration algorithms assessment: Bronze Standard application enactment on grids using the MOTEUR workflow engine

Tristan Glatard, Johan Montagnat, Xavier Pennec

Medical image registration algorithms assesment: Bronze Standard application enactment on grids using the MOTEUR workflow engine.

Tristan Glatard[1], Johan Montagnat[1], Xavier Pennec[2]
[1] CNRS, I3S laboratory, http://www.i3s.unice.fr/~johan
[2] INRIA Sophia Antipolis, http://www-sop.inria.fr/

Abstract

Medical image registration is pre-processing needed for many medical image analysis procedures. A very large number of registration algorithms are available today, but their performance is often not known and very difficult to assess due to the lack of gold standard. The Bronze Standard *algorithm is a very data and compute intensive statistical approach for quantifying registration algorithms accuracy.*

In this paper, we describe the Bronze Standard application and we discuss the need for grids to tackle such computations on medical image databases. We demonstrate MOTEUR, a service-based workflow engine optimized for dealing with data intensive applications. MOTEUR eases the enactment of the Bronze Standard and similar applications on the EGEE production grid infrastructure. It is a generic workflow engine, based on current standards and freely available, that can be used to instrument legacy application code at low cost.

1 The Bronze Standard application

Computerized medical image analysis is now a well established area that provides assistance for diagnosis, modeling, and pathologies follow-up. With the growing inspection capabilities of imagers and the medical data production growth, the need for large amounts of data storage and computing power increases. Grids have been identified as a tool suitable for dealing with medical data. Successful example of grid application deployment for image databases analysis, optimization of medical image algorithms, simulation, etc, have already been reported [7].

1.1 Medical images registration

Medical image registration algorithms are playing a key role in a very large number of medical image analysis procedures. Together with image segmentation algorithms, they are fundamental processings often needed prior to any subsequent analysis. Image registration consists in searching a 3D transformation between two images, so that the first one can superimpose on the second one in a common 3D frame. The transformation may be *rigid* (the composition of a translation and a rotation) to express a 3D change of frame or *non rigid* to express local deformations of space. A rigid registration is useful for aligning similar data (such as images of a same patient acquired at different times) into a single frame. A non-rigid registration is useful for computing the deformation map between different data (such as data acquired from two different patients). In addition, the registration

is said *mono-modal* when both images have been acquired using the same imaging modality (thus sharing some common signal characteristics) or *multi-modal* when the modalities differ (signal differences have then to be compensated for).

The computational load of these algorithms greatly varies depending on the type of registration computed, the size of the images to process, and the algorithms themselves. In general non-rigid, multi-modal algorithms are more costly than rigid, mono-modal algorithms. On typical 3D images and using up-to-date PCs, the computation time varies from a few minutes in the simplest cases to tens of hours in most compute intensive registrations.

1.2 Registration algorithms assessment

Given the very common use of registration algorithms and the different contexts for their application, a large number of new algorithms is developed by the research community. There are approximately a hundred of new research papers published on that subject each year. A difficult problem, as for many other medical image analysis procedures, is the assessment of these algorithms robustness, accuracy and precision [4]. Indeed, there is no well established *gold standard* to compare to the algorithm results. Different approaches have been proposed to solve this issue. It is possible to synthesize images by simulating the acquisition physics and to experiment the algorithm on the synthetic images produced [1]. However, realistic images are difficult to produce and hardly perfect enough for fine assessment of the algorithms. Phantoms (manufactured objects with properties close to human tissues for the imaging modality studied) can also be used to acquire test images. However, it is also very difficult to manufacture realistic enough phantoms.

1.3 The Bronze Standard method

An alternative for assessing registration algorithms is a statistical approach called the *Bronze Standard* [9]. The goal is basically to compute the registration of a maximum of image pairs with a maximum number of registration algorithms so that we obtain a largely overestimated system to relate the geometry of all the images. It makes this application very compute and data-intensive.

Suppose that we have n images of the same organ of one patient and m registration algorithms. We have in fact only $n-1$ free transformations to estimate that relate all these images, say $\bar{T}_{i,i+1}$. The transformation between images i and j is obtained using a compositions such as $\bar{T}_{i,j} = \bar{T}_{i,i+1} \circ \bar{T}_{i+1,i+2} \circ \ldots \circ \bar{T}_{j-1,j}$ if $i < j$ (or the inverse of both terms if $j > i$). The free transformation parameters are computed by minimizing the prediction error on the observed registrations:

$$\min_{\bar{T}_{1,2},\bar{T}_{2,3},\ldots,\bar{T}_{n-1,n}} \sum_{i,j\in[1,n],k\in[1,m]} d\left(T^k_{i,j}, \bar{T}_{i,j}\right)^2 \tag{1}$$

where $T^k_{i,j}$ is the transformation computed between image i and j by the k^{th} registration algorithm, and d is a distance function between transformations chosen as a robust variant of the left invariant distance on rigid transformation [11]. The estimation $\bar{T}_{i,i+1}$ of the perfect registration $T_{i,i+1}$ is called bronze standard because the result converges toward $T_{i,i+1}$ as the number of methods m and the number of images n become larger. Indeed, considering a given registration method, the variability due to the noise in the data decreases as the number of images n increases, and the registration computed converges toward the perfect registration up to the intrinsic bias (if there is any) introduced by the method. Now, using different registration procedures, based on different methods, the intrinsic bias of each method also becomes a random variable, which is hopefully centered around

zero and averaged during the minimization procedure. The different biases of the methods are now integrated into the transformation variability. To fully reach this goal, it is important to use as many independent registration methods as possible.

In this process, we do not only estimate the optimal transformations, but also the rotational and translational variance of the "transformation measurements", which are propagated through the criterion to give an estimated of the variance of the optimal transformations. These variances should be considered as a fixed effect (i.e. these parameters are common to all patients for a given image registration problem, contrarily to the transformations) so that they can be computed more faithfully by multiplying the number of patients.

An important variant of the Bronze Standard is to relax the assumption of the same variances for all algorithms, and to unbias their estimation. This can be realized by using only $m-1$ out of the m methods to determine the bronze standard registration, and use the obtained reference to determine the accuracy of the last method.

In this paper, we are considering $m = 4$ different registration algorithms in our implementation of the bronze standard method: (1) **Baladin** and (2) **Yasmina** are intensity-based. The former uses a block matching strategy while the later optimizes a similarity measure on the complete images using the Powel algorithm. (3) **CrestMatch** is a prediction-verification method and (4) **PFRegister** is based on the ICP algorithm. Both CrestMatch and PFRegister register features (crest lines) extracted from the input images. These algorithms are further described in [9]. Figure 1 illustrates the application workflow. Each box in figure 1 represents an algorithm and arrows show computation dependencies.

2 Enacting the application workflow on the EGEE production grid

Even though registration computations are usually tractable on simple PCs, the large number of input data and registration algorithms needed to compute the bronze standard makes this method very compute intensive. A grid infrastructure can handle the load of the computations involved and help in managing the medical image database to process.

2.1 EGEE infrastructure

In order to evaluate the relevance of our prototype and to compare real executions to theoretically expected results, we made experiments on the EGEE production grid infrastructure[1]. This platform is a pool of thousands computers (standard PCs) and storage resources accessible through the LCG2 middleware[2]. The resources are assembled in computing centers, each of them running its internal batch scheduler. Jobs are submitted from a user interface to a central Resource Broker which distributes them to the resources available. On such a grid infrastructure, the application parallelism can be exploited to optimize the execution time. Several instances of each service will be concurrently submitted to the grid and executed on different processors.

2.2 Application workflow

The Bronze Standard application is composed as a workflow of algorithms represented on figure 1. The two input image sources on top correspond to the image

[1] Enabling Grids for E-sciencE, http://www.eu-egee.org

[2] LCG2 middleware, http://lcg.web.cern.ch/LCG/activities/middleware.html

sets on which the evaluation is to be processed. The upper box corresponds to an initialization needed for the registration algorithms. Then come the registration algorithms themselves and format conversion and result collection services. Finally, the bottom (gray) service is responsible for the evaluation of the accuracy of the registration algorithms, leading to the outputs values of the workflow. It computes means from all the results of the registration services considered but one, and evaluates the accuracy of the specified registration method. This service has to be synchronized: it must be enacted only once every data have been processed in the workflow. The six services with a triple contour are compute intensive initialization and registration algorithms while the other boxes represent more lightweight computation steps such as data format transformations.

2.3 Medical workflows

Similarly to the Bronze Standard application presented above, medical image analysis procedures are often not based on a single image processing algorithm but rather assembled from a set of basic tools dedicated to process the data, model it, extract quantitative information, and analyze results. Given that interoperable algorithms packed in software components with a standardized interface enabling data exchanges are provided, it is possible to build complex workflows to represent such procedures for data analysis. High level tools for expressing and handling the computation flow are therefore expected to ease computerized medical experiments development.

When dealing with medical experiments, the user often needs to process datasets made of *e.g.* hundreds of individual images. The workflow management is therefore data driven and the scheduler responsible for sharing the load of computations should take into account the input data sets as well as the workflow graph topology.

3 MOTEUR workflow engine

We implemented an hoMe-made OpTimisEd scUfl enactoR (MOTEUR) prototype to manage application workflows. MOTEUR is written in Java, in order to be platform independent. It is available under CeCILL Public License (a GPL-compatible open source license) at `http://www.i3s.unice.fr/~glatard`. The workflow description language adopted is the Simple Concept Unified Flow Language (Scufl) used by the Taverna workbench [10].

Figure 1 shows the MOTEUR web interface representing a workflow that is being executed. Each service is represented by a color box and data links are represented by curves. The services are color coded depending on their current status: gray services have never been executed; green services are running; blue services have finished the execution of all input data available; and yellow services are not currently running but waiting for input data to become available.

MOTEUR is interfaced to the job submission interfaces of both the EGEE infrastructure and the Grid5000[3] experimental grid. In addition, lightweight jobs execution can be orchestrated on local resources. MOTEUR is able to submit different computing tasks on different infrastructures during a single workflow execution.

3.1 Service-based approach

To handle user processing requests, two main strategies have been proposed and implemented in grid middlewares:

[3]Grid5000, `http://www.grid5000.org`

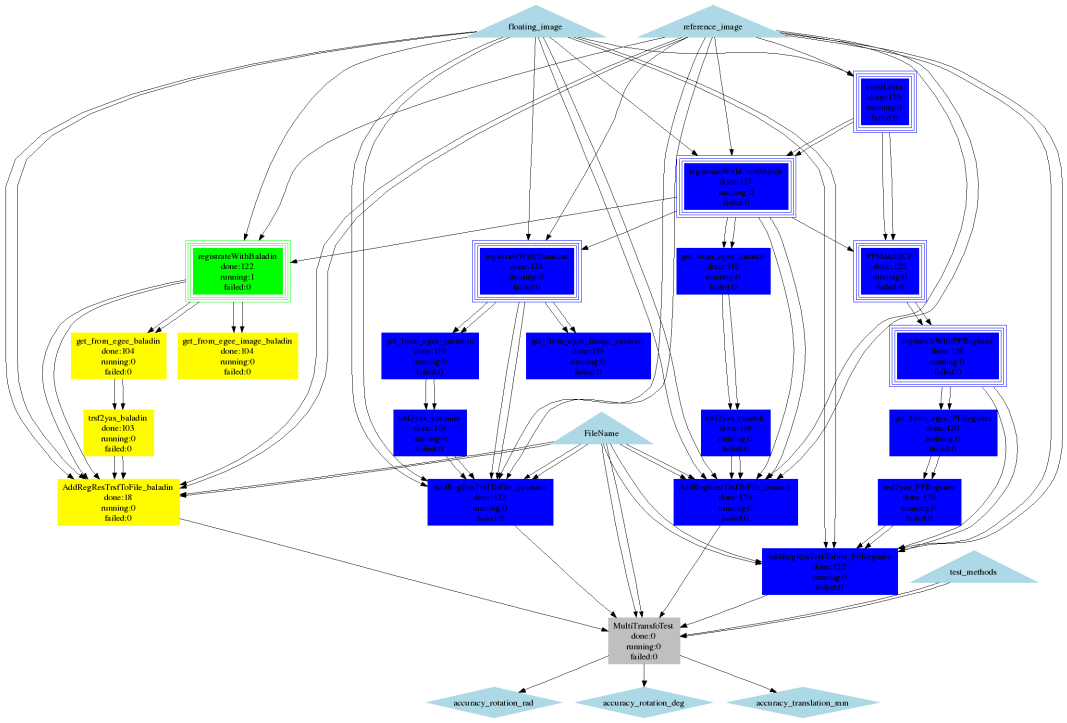

Figure 1: MOTEUR interface representation

1. In the *task based* strategy, also referred to as *global computing*, users define computing tasks to be executed. Any executable code may be requested by specifying the executable code file, input data files, and command line parameters to invoke the execution. The task based strategy, implemented in GLOBUS [3], LCG2 or gLite [4] middlewares for instance, has already been used for decades in batch computing. It makes the use of non grid-specific code very simple, provided that the user has a knowledge of the exact syntax to invoke each computing task.

2. The *service based* strategy, also referred to as *meta computing*, consists in wrapping application codes into standard interfaces. Such services are seen as black boxes from the middleware for which only the invocation interface is known. The services paradigm has been widely adopted by middleware developers for the high level of flexibility that it offers. However, this approach is less common for application code as it requires all codes to be instrumented with the common service interface.

The service-based approach is naturally very well suited for chaining the execution of different algorithms assembled to build an application. Indeed, the interface to each application component is clearly defined and the middleware can invoke each of them through a single protocol. In addition, the service-based approach offers a large flexibility for managing applications requiring the processing of complete image databases such as the Bronze Standard described above. The input data are treated as input parameters, and the service appears to the end user as a black box hiding the code invocation.

[4] gLite middleware, http://www.glite.org

When a service is dealing with two input data sets or more, the semantics of the service with regard to the data composition needs to be specified. MOTEUR implements two data composition patterns:

- The one-to-one composition: each input of the first data set $\{A\}_{i\in[1,m]}$ is processed with each input of the second data set $\{B\}_{i\in[1,n]}$, thus producing $\min(m,n)$ output data.
- The all-to-all composition: all input of $\{A\}_{i\in[1,m]}$ are processed with all input of $\{B\}_{i\in[1,n]}$, thus producing $m \times n$ output data.

The use of these two composition strategies, embedded in the Scufl language, significantly enlarges the expressiveness of the workflow language. It is a powerful tool for expressing complex data-intensive processing applications in a very compact format.

MOTEUR is implementing an interface to both Web Services [13] and GridRPC [8] application services. We developed an XML-based language to be able to describe input data sets. This language aims at providing a file format to save and store the input data set in order to be able to re-execute workflows on the same data set.

3.2 Enabling legacy codes

In the service based approach, all application codes need to be wrapped into a standard service envelope. This increases the code complexity on the application developer side and this prevent the use of legacy code which cannot necessarily be modified and recompiled for various reasons.

In order to face this limitation, we have developed a legacy code application wrapping service similar to GEMLCA [5]. The idea is to propose a standard web service capable of submitting any legacy executable on the target grid infrastructure. This generic application service, is dynamically composing the executable invocation command line before submission. For this purpose, it needs a description of the executable command line parameters. We have defined a simple XML-based parameters description format. For each legacy code to gridify, the user only needs to produce the corresponding XML document. The generic service is taking as input both the executable and the description document.

The generic application service is installed on the grid user interface and it does not require any deployment on the grid computing resources. It submits jobs to the grid through the standard workload management system.

3.3 Optimizing the execution of data intensive applications

Some workflow managers, such as the CONDOR DAGMan [5] have adopted the task-based approach, coupling processings and data to be processed. This static and complete description of the graph of tasks to be executed eases the optimization of the workflow execution as it provides all information necessary for mapping the workflow and data to available resources (see for instance the Pegasus system [2]). However, it poorly deals with large data sets since a new task need to be explicitly written for each input data to be processed.

In service-based workflow managers such as MOTEUR, Kepler [6], Taverna [10] or Triana [12], each processor is invoking external services whose data is dynamically transmitted as parameter. However, the services invocation is an extra layer between the workflow manager and the execution grid infrastructure. The workflow manager has no direct access to the grid resources and therefore it cannot directly optimize the job submissions scheduling. Performances are critical in the case

[5] CONDOR DAGMan, http://www.cs.wisc.edu/condor/dagman

of data-intensive applications and MOTEUR is implementing several optimization strategies to ensure optimal workflow execution by exploiting the massively parallel resources available on the grid infrastructure.

Workflow parallelism. The workflow encompasses an inherent degree of parallelism as several independent services may be invoked in parallel asynchronously by the workflow engine.

Data parallelism. The computations described in the workflow can be performed independently for each input data segment. When dealing with large input data sets, this is a considerable potential optimization that consists in processing all these data in parallel on different grid resources. Also quite obvious, the data parallelism is not straight forward to implement. Indeed, parallel execution over different data leads to loose computation sequences (a data can overtake another one in the workflow) and potential causality problem if the ordering is not reestablished. MOTEUR' strategy to avoid this problem is to associate to each processed data segment a complete history tree of the former processings that unambiguously describes the data provenance. To deal with the all-to-all composition strategy, MOTEUR also keeps in memory all data segments sent to the input of each service. Thus, when a delayed data arrives it can be composed with all formerly identified input data by repetitive invocations of the service.

Services parallelism. The computations of different services over different input data sets can overlap in time. Parallel computing of such tasks enables a pipelining optimization similar to the one exploited inside CPUs. Theoretically, this service parallelism should not bring an extra level of parallelism when data parallelism is exploited. If all data could be processed in parallel in constant time, there would be no overlap of successive services. In practice though, execution times on a loaded production infrastructure are highly variable and unpredictable. The desynchronization of the computations creates the need for service parallelism optimization.

Jobs grouping. Finally, sequential jobs might be grouped and executed to lower the number of services invocation and minimize the grid overhead resulting from jobs submission, scheduling and data transfers. Jobs grouping is not feasible in general on a service-based infrastructure as services are completely independent and can only be invoked separately by the workflow engine. The internal logic of all services implemented through the generic wrapping service is known though. The workflow engine is thus capable of translating the calls to two consecutive generic services into a call to a single service submitting a compound job with two consecutive executable command line invocations.

To our knowledge, MOTEUR is the first service-based workflow manager implementing all these levels of parallelism.

4 Results and conclusions

MOTEUR is evaluated on the Bronze Standard application with a realistic experimental setting. We executed our workflow on different inputs data sets, with various sizes. Input image pairs are taken from a database of injected T1 brain MRIs from the cancer treatment center "Centre Antoine Lacassagne" in Nice, France, courtesy of Dr Pierre-Yves Bondiau. All images are $256 \times 256 \times 60$ and coded on 16 bits, thus leading to a 7.8 MB size per image. Each of the input image pair was registered

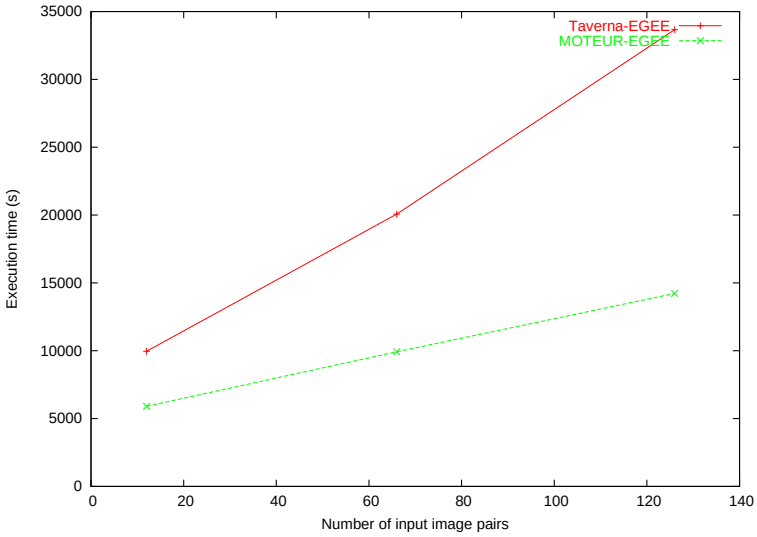

Figure 2: Execution times of MOTEUR vs Taverna on the EGEE production infrastructure

with the 4 algorithms and leads to 6 grid job submissions (triple contour services in figure 1). The 4 rigid registration algorithms used reached a sub-voxel accuracy of 0.15 degree in rotation and 0.4 mm in translation for the registration of these images.

4.1 MOTEUR performances

The first experiment, reported in figure 2, is a comparison of MOTEUR performances against the Taverna workflow manager [10]. Taverna is a service-based workflow manager targeting bioinformatics application that is being developed in the UK eScience MyGrid project. Taverna has become a reference workflow manager in the eScience community. The figure displays the execution times obtained with Taverna and MOTEUR w.r.t. the number of input data sets. The figure shows that MOTEUR introduces an average speed-up of 2.03. Even more interesting, this speed-up is growing with the number of input data sets to process. The performance gain is due to the full exploitation of the data and services parallelism: Taverna does not provide service parallelism and data parallelism is limited to a fixed number of parallel invocations.

The second experiment reported in figure 3 quantifies the performance gain introduced by the different level of optimization implemented in MOTEUR. We executed the Bronze Standard workflow on 3 different inputs data sets composed by 12, 66 and 126 image pairs, corresponding to images from 1, 7 and 25 patients respectively. In total, the workflow execution resulted in 6 times more job submissions (72, 396 and 756 jobs respectively). We computed the Bronze Standard with different optimization configurations in order to identify the specific gain provided by each optimization.

The reference curve (plain curve, labeled NOP) corresponds to a naive execution where only workflow parallelism is activated. The Job Grouping optimization (JG curve) reduces the jobs submission overhead as expected. The time gain is

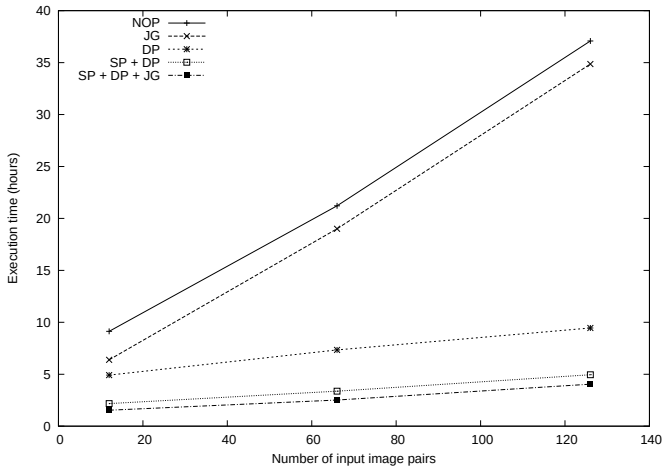

Figure 3: Comparison of the execution times obtained for different optimization configurations

almost constant independently of the number of input data. Unsurprisingly, the most drastic optimization is the Data Parallelism (DP curve) for this data intensive application. The speed-up grows with the number of images to be processed (the DP curve slope is lower than the reference curve slope). Theoretically, the DP curve should be horizontal (no overhead introduce by the increasing number of data) given that the number of grid processing units exceeds the number of jobs submitted. However, the EGEE grid is exploited in production mode (24/7 workload) by a large multi-users community. Therefore, the Service Parallelism optimization (DP+SP curve) further improves performances. Finally, combining all these optimizations (SP+DP+JG curve) provides the best result. The final speed-up is higher than 9.1 when considering the largest scale experiment.

4.2 Conclusions

Data intensive applications are common in the medical image analysis community and there is an increasing need for computing infrastructures capable of efficiently processing large image databases. The Bronze Standard application is a concrete example to registration algorithms assessment with an important impact for medical image analysis procedures. The application is assembled from a set of legacy code components, wrapped into a generic web service and enacted on the EGEE grid through the MOTEUR workflow enactor.

We demonstrated MOTEUR capabilities and performances. This workflow engine is conforming to the Scufl workflow description language. It implements interfaces to Web and GridRPC services. MOTEUR has been interfaced to the EGEE production grid infrastructure and the Grid5000 experimental infrastructure. The workflow execution is optimized using different parallelization strategies enabling the exploitation of the grid parallel resources. MOTEUR is freely available for download under a GPL-like license.

Acknowledgments

This work is partly funded by the French research program "ACI-Masse de données" (http://acimd.labri.fr/), AGIR project (http://www.aci-agir.org/). We are grateful to the EGEE European IST project (http://www.eu-egee.org) for providing the infrastructure used in the experiments presented.

References

[1] H. Benoit-Catttin, F. Bellet, J. Montagnat, and C. Odet. Magnetic Resonance Imaging (MRI) Simulation on a Grid Computing Architecture. In *Biogrid'03, proceedings of the IEEE CCGrid03*, Tokyo, Japan, May 2003.

[2] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, and G. Mehta et al. Mapping abstract complex workflows onto grid environments. *Jnl of Grid Comp.*, 1(1):9 – 23, 2003.

[3] Ian Foster. Globus Toolkit Version 4: Software for Service-Oriented Systems. In *International Conference on Network and Parallel Computing (IFIP)*, volume 3779, pages 2–13. Springer-Verlag LNCS, 2005.

[4] P. Jannin, J.M. Fitzpatrick, D.J. Hawkes, X. Pennec, R. Shahidi, and M.W. Vannier. Validation of medical image processing in image-guided therapy. *IEEE Trans. on Medical Imaging*, 21(12):1445–1449, December 2002.

[5] Pter Kacsuk, Ariel Goyeneche, Thierry Delaitre, Tams Kiss, Zoltn Farkas, and Tams Boczko. High-Level Grid Application Environment to Use Legacy Codes as OGSA Grid Services. In *Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing (GRID '04)*, pages 428–435, Washington, DC, USA, 2004. IEEE Computer Society.

[6] Bertram Ludscher, Ikay Altintas, Chad Berkley, Dan Higgins, Efrat Jaeger, Matthew Jones, Edward A. Lee, Jing Tao, and Yang Zhao. Scientific Workflow Management and the Kepler System. *Concurrency and Computation: Practice & Experience*, 2005.

[7] J. Montagnat, F. Bellet, H. Benoit-Cattin, V. Breton, L. Brunie, H. Duque, Y. Legré, I.E. Magnin, L. Maigne, S. Miguet, J.-M. Pierson, L. Seitz, and T. Tweed. Medical images simulation, storage, and processing on the european datagrid testbed. *Journal of Grid Computing*, 2(4):387–400, December 2004.

[8] Hidemoto Nakada, Satoshi Matsuoka, K Seymour, J Dongarra, C Lee, and Henri Casanova. A GridRPC Model and API for End-User Applications. Technical report, Global Grid Forum (GGF), jul 2005.

[9] Stphane Nicolau, Xavier Pennec, Luc Soler, and Nicholas Ayache. Evaluation of a New 3D/2D Registration Criterion for Liver Radio-Frequencies Guided by Augmented Reality. In *International Symposium on Surgery Simulation and Soft Tissue Modeling (IS4TM'03)*, volume 2673 of *LNCS*, pages 270–283, Juan-les-Pins, 2003. INRIA Sophia Antipolis, Springer-Verlag.

[10] Tom Oinn, Matthew Addis, Justin Ferris, Darren Marvin, Martin Senger, Mark Greenwood, Tim Carver, Kevin Glover, Matthew R. Pocock, Anil Wipat, and Peter Li. Taverna: A tool for the composition and enactment of bioinformatics workflows. *Bioinformatics journal*, 17(20):3045–3054, 2004.

[11] X. Pennec, R.G. Guttman, and J.-P. Thirion. Feature-Based Registration of Medical Images: Estimation and Validation of the Pose Accuracy. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI'98)*, volume 1496 of *LNCS*, pages 1107–1114, Cambridge, USA, October 1998. Springer.

[12] Ian Taylor, Ian Wand, Matthew Shields, and Shalil Majithia. Distributed computing with Triana on the Grid. *Concurrency and Computation: Practice & Experience*, 17(1–18), 2005.

[13] (W3C) World Wide Web Consortium. Web Services Description Language (WSDL) 1.1, mar 2001.