Brill Academic Publishers
P.O. Box 9000, 2300 PA Leiden,
The Netherlands

Computing Letters (CoLe)
vol. 1, no. 3, 2005, pp. 85-91
ISSN 1574–0404

On Deploying Scientific Software within the Grid-TLSE Project

Eddy Caron, Frédéric Desprez
LIP-ENS Lyon/INRIA, 46 allée d'Italie, 69364 Lyon Cedex 07, France
{Eddy.Caron, Frederic.Desprez}@ens-lyon.fr,
tel +(33) 472 728 569 — fax +(33) 472 728 080

Michel Daydé, Aurélie Hurault[1], Marc Pantel
FERIA-IRIT-ENSEEIHT, 2 rue Camichel, 31071 Toulouse Cedex, France
{Michel.Dayde, Aurelie.Hurault, Marc.Pantel}@enseeiht.fr,
tel +(33) 561 588 270 — fax +(33) 561 588 306

Received 10 May, 2005; accepted in revised form 15 July, 2005

Abstract: The main goal of the Grid-TLSE project is to design an expert site that provides an easy access to a number of sparse matrix solver packages allowing their comparative analysis on user-submitted problems, as well as on matrices from collections also available on the site. The site provide user assistance in choosing the right solver for its problems and appropriate values for the solver parameters. A computational Grid is used to deal with all the runs arising from user requests.
After an overview of the project, we discuss on some of the main difficulties we face in Grid-TLSE: considering the amount of softwares that will be available, facilitating to facilitate their deployment and their exploitation over a Grid is crucial. We make use of an *abstract meta-data based description of solvers* and have introduced the concept of *scenario*, i.e. a high-level user assistance description that provides ways of discovering automatically the most appropriate solvers based on their description. The scenarios are used for generating the dynamic workflows that will be executed over the Grid and take advantage of some of the features available in the DIET Grid middleware. We also report on the introduction of a semantic-based component model for efficient service trading.

Keywords: Sparse solvers, Computational Grid, DIET middleware, Metadata, Service trading

1 Goals of the project

The Grid-TLSE project is a three-year project that has started in January 2003 and which is funded by the French Ministry of Research ACI GRID Program ([1], [2], [5]).

The academic partners involved in this project are: CERFACS (Toulouse), IRIT (Toulouse), LaBRI (Bordeaux) and LIP-ENSL (Lyon). These teams have been working together over many years in the field of sparse matrix computations and have a strong collaboration with the international scientific community working in this domain that has given rise to the production of several software packages available to external users. Strong of this experience, these research teams have decided to build a web site giving an easy access to tools and allowing comparative synthetic analysis of these packages. Industrial partners (CEA, CNES, EADS, EDF, IFP) are involved in the project and help in specifying the main features of the expert site.

The user can obtain synthetic statistics from actual runs of a variety of sparse matrix solvers on chosen matrices or can interrogate the databases available for information and references related

[1]The work of this author is part of her PhD thesis.

to sparse linear algebra. He can either submit his/her own problem, or use a matrix from the database available on the site including public domain matrix collections such as the Rutherford-Boeing collection ([13]), the University of Florida sparse matrix collection ([11]), ...

Each request to the Grid-TLSE site may lead to a large number of computations. When comparing several sparse matrix packages using several values of the control parameters over different matrices, most runs are independent and can be executed simultaneously. Even if these runs can be conveniently handled by a sufficiently large Grid an important part of our work is to design a sophisticated expert analyser to limit the amount of elementary computational requests sent to the Grid. This analyser will rely on scenarios designed by sparse solver experts and on metadata describing these solvers.

2 Structure of the Grid-TLSE site

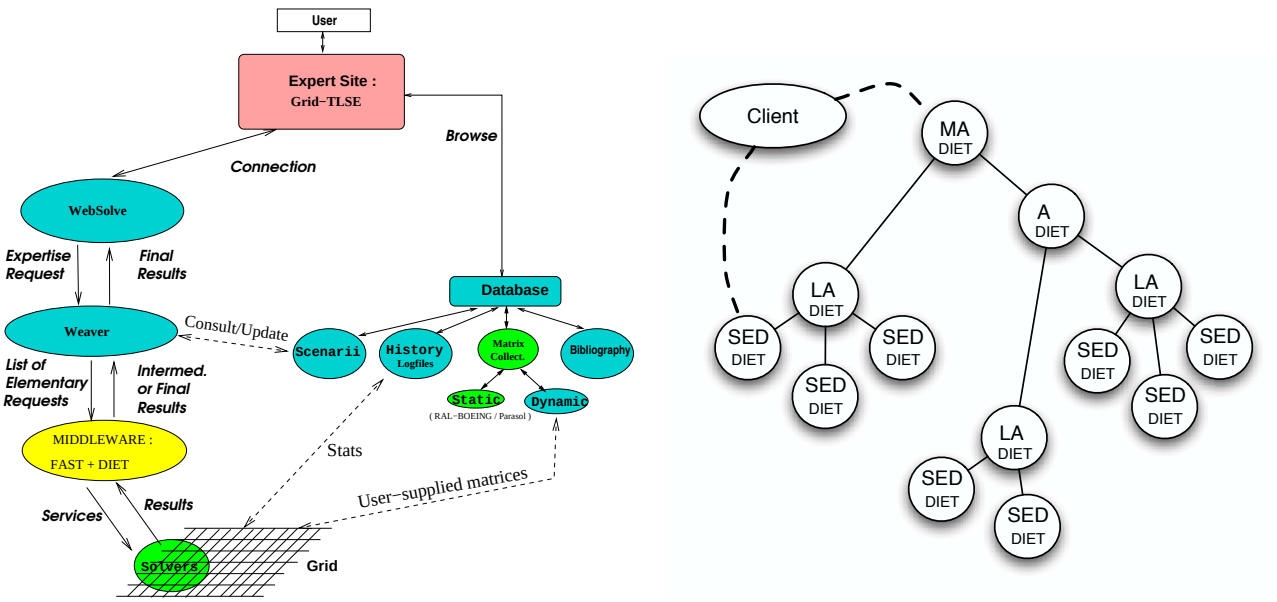

Figure 1: Structure of the Grid-TLSE web site (left) and DIET hierarchical organization (right).

The Figure 1 (left) describes the main components of the Grid-TLSE site and their relationships:

WebSolve - a Web interface for computations and user assistance over a Grid. It allows a user using a standard WWW navigator to submit computational or assistance requests to a Grid, to browse the matrix database, to upload/download a matrix, to monitor the submitted requests, to manage and add solvers and scenarios and to check for their correct execution.

Weaver - an infrastructure for user assistance over a Grid. It converts a general assistance request into sequences of elementary solver runs. It is also in charge of the deployment and the exploitation of services over a Grid through the DIET middleware.

DIET - a middleware to access the computational Grid ([7]).

Database - a database that stores all necessary data to efficiently construct the sequence of solver runs corresponding to an assistance request. In particular, it contains all the metadata describing solvers, computers, matrices and user assistance scenarios.

One of the main research issue in the project is the specification of the procedures for the user assistance and the management of the site. In particular, we have designed a framework

for the description and management of solvers and scenarios, and have developed procedures for: adding new software packages, definition of new scenarios by expert users, exploiting the computed statistics i.e. being able to "reuse" results, avoid repetition of runs, and study the typical behaviour of a solver or the properties of a matrix.

3 The DIET middleware

The DIET Grid middleware [7] and associated tools (FAST, GoDIET, LogService, VizDIET) are developed within the framework of the GRID-ASP project by LIP-GRAAL (partner of the project) and LIFC-SRDP. In particular it provides facilities for defining the service profile (typed list of in/inout/out parameters) as well as a scalable scheduling hierarchy. We also exploit DIET plug-in schedulers to adapt the scheduling part of the tools to better fit the characteristics of our application.

3.1 Overview of DIET

The GridRPC paradigm [14] is a efficient approach for client-server computing over the Grid. Clients submit computation requests to a scheduler that aims at finding the most efficient server available on the Grid. Scheduling is used for balancing the work among the servers. A list of best available servers is sent back to the client which in turn sends the data and the request to solve a given problem. Due to the growth of the network bandwidth and the reduction of the latency, even small computation requests can be sent to the servers available on the Grid. As a consequence, the scalability of the middleware itself is becoming a crucial issue. The scheduling can be made scalable by distributing the scheduler.

DIET [3] is designed as a set of hierarchical elements (Client, Master Agent – MA –, Local Agent – LA –, and server daemon – SeD –) to build applications following the GridRPC paradigm at the client level. This middleware is able to find an appropriate server according to the information involved in the client request (problem to be solved, size of the data involved), the performance of the target platform (server load, available memory, communication performance), and the availability of data stored during previous computations. The scheduler is distributed using several hierarchies connected either statically (in a Corba fashion) or dynamically (in a peer-to-peer fashion). Figure 1 (right) shows an example of a DIET hierarchy

3.2 Recent developments in DIET

Several features have been added since the last version of DIET.

First a tool has been developed for the automatic deployment of the platform based on description file written in XML (GoDIET). An environment for the monitoring (LogService) and visualisation (VizDIET) has been developed for understand the behaviour of the DIET system on a Grid platform.

Data management is an important problem that needs to be solved efficiently to avoid too much overhead. Current version of DIET uses DTM developed at Besançon [6]. It uses a hierarchical approach mapped on the DIET hierarchy.

Finally, we are working on plugin schedulers. This allows the user to play with the internals of agents and tune DIET's scheduling by changing the heuristics, adding queues, changing the performance metrics, and the aggregation functions designed to take into account many performance metrics and how to combine them, ... We believe that these features will be useful both for computer scientists to test their algorithms on a real platform and expert application scientists to tune DIET for specific application behaviours.

4 Grid-TLSE user assistance scenarios

A typical Grid-TLSE user assistance request will ask for the performance of some solvers on several matrices according to some metrics. This will require a significant amount of solver runs which

can easily be reduced using insights provided by sparse solver experts. Most of the time, users will require more precise assistance than figures comparing simple solver runs. Again, this could be provided by experts in sparse linear algebra. This is the purpose of user assistance scenarios which provide users precise synthetic answers while reducing the combinatorial nature of the runs. Therefore, an assistance request will also contain an objective (scenario) required by the user. In order to ease the description of scenarios, their structure is hierarchical : that is, a scenario can use others scenarios.

As an example, the objectives (scenarios) available in the first version of the Grid-TLSE web site are:

1. **Solve**: to use a solver with its default parameter to solve a problem,
2. **Threshold Sensitivity**: to study the quality of the numerical solution as a function of the pivoting threshold,
3. **Ordering Sensitivity**: to evaluate the behaviour of solvers depending on the ordering heuristic applied,
4. **Minimum Time**: to estimate which combination of solver/ordering leads to the smallest computation time on a given problem.

We now give some details about the "Ordering sensitivity" scenario to illustrate its hierarchical structure (an ordering is a heuristic to permute the graph of the initial matrix with the aim to limit the cost of the numerical factorisation; the ordering has a strong impact on both the number of operations and memory used by a solver):

Phase 1 (sub-scenario): Get orderings. If only one solver has been specified by the user: get all its internal orderings, if more than one solver has been specified: get all possible orderings from all solvers.

Phase 2 (sub-scenario): Obtain the values of the required metrics for each ordering. For metrics of type "estimated": only the analysis is performed for each required solver, for metrics of type "effective": the factorisation is also performed.

Phase 3: Report metrics for all combinations of solvers/orderings.

This scenario is static: the set of required experiments in order to perform the user assistance does not depend on the results of some of the experiments. Scenarios can also be dynamic, i.e. the results of some preliminary experiments are used to derive subsequent experiments. For example, the **Minimum Time** scenario uses the **Ordering sensitivity** scenario in order to produce all potential pair of ordering/solver and compute the estimated execution through a low cost symbolic analysis. It then selects the best ordering for each solver and produces a new set of experiments in order to compute the effective execution time and finally reports the user the best ordering/solver pair for its problem.

In order to provide assistance or reduce combinatorial costs, a scenario can use solver, matrix or computer features. A given scenario will therefore require some features from the solvers it aims at comparing. Grid-TLSE should therefore provide a framework for the description of solver's features.

5 Grid-TLSE meta-data framework for sparse solvers

Sparse direct solvers are quite similar since they all solve a sparse linear system using some computer resources and provide a result with a given numerical precision. But they are also very different in practice since they all use different algorithms with their own control parameters.

Since the main purpose of the project is to help the user in choosing the right solver and an appropriate selection of parameter values, we should manage to proceed to a comparison of the

solvers. As a consequence, all solvers should provide a similar interface to the scenarios, and all scenarios should provide a similar interface to the client. One simple approach is to write wrappers around all the solvers that take the same parameters and produce the same results. This can be quite heavy in term of development cost and is very sensitive to the addition of new parameters that may require the modification of all wrappers. Note that the interface common to all solvers may also be very poor.

A more interesting approach relies on the use of meta-data which describe all the possible parameters and results (and their possible values) for each solver (see [9, 10] for a more detailed description of Grid-TLSE metadata framework). Given a set of solvers, it is then possible to compute the intersection of their parameters (and the parameter values) which will offer more possibilities than the common interface approach.

The meta-data approach allows the expert to define all the possible parameters and to define each solver according to the meta-data. The scenario approach allows the expert to define scenario using the meta-data and to define the graphics returned back to the user as a result of the assistance request.

The experiments are then represented as a set of meta-data values. This set will be transmitted through the middleware to a minimal wrapper translating meta-data values to real solver parameters and results.

6 Semantic based component model for linear algebra service trading

The services that are deployed within the project are written with different languages (C, C++, F77, F90). There are hundreds of services of many type: solver, matrix validation, matrix generators, ordering and scaling generators, ... Additionally, the same service (i.e. the same piece of software) can be available on different computers possibly in different releases.

One of the main challenge when using a Grid providing a large range of softwares is finding the most appropriate software packages and computer architecture for a given user request. The solver meta-data can be used in order to improve the trading process which is usually based on the service interface. However the semantics contained in the interface and the meta-data are still very poor.

Within the Grid-TLSE Project, each service can be defined as a composition of linear algebra operators. An user request can be defined in the same way. We then use the mathematical properties of linear algebra operators (associativity, commutativity, distributivity, null or neutral element, ...) in order to select the best service or composition of services for processing an user request.

This approach is based on e-unification, that is, unification modulo an equational theory (the properties of the linear algebra operators). We propose to use an energy-sensitive heuristic which we have proven to be complete (given an unbound quantity of energy). This heuristic will interact with the DIET middleware in order to choose the best solution according to static and dynamic criteria (see [8] for a more detailed description).

7 Conclusion

We have considered some of the main issues developers may face when designing a Grid-based project that requires handling a wide range of complex scientific softwares. For facilitating the deployment and the exploitation of these softwares, our approach rely on the use of an abstract description of the softwares and on the notion of scenario which can be seen as a high level description of the operations that are required for a given expertise. These scenarios are then used for dynamically generating workflows. As a consequence, they do not require modification when adding or removing a software since softwares are automatically discovered. Similarly, the descriptions of solvers do not require any modification when adding or removing a scenario.

Our Grid infrastructure uses the DIET GridRPC middleware developed at LIP-ENSL. Its recent version provides new functionalities that allow for a more efficient management of the

computational Grid: GoDIET for the automatic deployment of the platform, and LogService and VizDIET that provides respectively monitoring and visualisation tools. The mechanism for managing persistency of internal data now available in DIET appears to be a crucial performance issue and we plan to use it in the next release of our software.

The current prototype demonstrates the main features of the expert site: an embryo of the WebSolve interface, some examples of expertise requests of increasing complexity , and the formats used to display the results (graph, bar-chart, tables). Since that, major modifications have been made to the WebSolve interface. Different types of users are defined (beginners, intermediates, advanced, experts, and administrators). We have also introduced the notion of groups of users, so that set of users can share matrices and data that are not public. Furthermore the expertise requests that were managed in a synchronous fashion are now managed asynchronously.

In the forthcoming version, the user will also be able to select the computing platforms within the ones available in the Grid-TLSE Grid. Some of the machines we will use are multiprocessors and are likely to only use batch. This motivates studies around scheduling – since we process quite complex workflows – and batch management. We are also working on the specification of the Weaver module, and in particular, on how to abstract the description of a sparse solver in a general way.

We expect that the TLSE web site will be widely open in Summer 2005. Additional information can be found at `http://www.enseeiht.fr/lima/tlse`.

References

[1] An overview of the Grid-TLSE project. Technical Report, http://www.enseeiht.fr/lima/tlse/refs.html, June 2002.

[2] P. Amestoy and M. Pantel. Grid-TLSE: A Web expertise site for sparse linear algebra. Sparse Days and Grid Computing in St Girons, France, June 10-13, 2003.

[3] E. Caron, F. Desprez, F. Lombard, J.-M. Nicod, M. Quinson, and F. Suter. A Scalable Approach to Network Enabled Servers. Proceedings of Euro-Par 2002, Paderborn, Germany, September 2002.

[4] E. Caron, F. Desprez, F. Petit, and C. Tedeschi. Resource Localization Using Peer-To-Peer Technology for Network Enabled Servers, Laboratoire de l'Informatique du Parallélisme (LIP), Research report, 2004-55, ftp://ftp.ens-lyon.fr/pub/LIP/Rapports/RR/RR2004/RR2004-55.pdf, 2004.

[5] M. Daydé, L. Giraud, M. Hernandez, J.-Y. L'Excellent, M. Pantel, and C. Puglisi, An Overview of the GRID-TLSE Project, Proceedings of 6th International Meeting VECPAR'04, 28-30 June 2004, Valencia, Spain, pp 851-856.

[6] B. Del-Fabbro, D. Laiymani, J.-M. Nicod, and L. Philippe. A Data Persistency Approach for the DIET Metacomputing Environment, Proceedings of International Conference on Internet Computing, Eds Hamid R. Arabnia, Olaf Droegehorn, and S. Chatterjee, CSREA Press, pp 701-707, Las Vegas, USA, June 2004.

[7] `http://graal.ens-lyon.fr/~diet/`.

[8] A. Hurault, M. Pantel, F. Desprez. Linear algebra service trader for GRIDs, Submitted to Euro-Par 2005.

[9] M. Pantel, C. Puglisi, P. Amestoy. Grid-TLSE metadata for solvers, matrices and computers, In Parallel Matrix Algorithms and Applications, Marseille (France), October 20-22, 2004.

[10] M. Pantel, C. Puglisi, P. Amestoy. GRID, components and scientific software, Submitted to Euro-Par 2005.

[11] http://www.cise.ufl.edu/research/sparse/matrices.

[12] M. Quinson. Dynamic Performance Forecasting for Network-Enabled Servers in a Metacomputing Environment, International Workshop on Performance Modelling, Evaluation, and Optimisation of Parallel and Distributed Systems (PMEO-PDS'02), April 15-19, 2002.

[13] http://www.cerfacs.fr/algor/Softs/RB.

[14] K. Seymour, C. Lee, F. Desprez, H. Nakada and Y. Tanaka. The End-User and Middleware APIs for GridRPC. Workshop on Grid Application Programming Interfaces, In conjunction with GGF12, Brussels, Belgium, September, 2004.

[15] R. Wolski and N. T. Spring and J. Hayes. The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing, Future Generation Computing Systems, Metacomputing Issue, Vol. 15, 5–6, pp 757–768, October, 1999.