

Legacy Code Support for Commercial Production Grids

Gabor Terstyanszky¹, Tamas Kiss¹, Peter Kacsuk^{1,2}, Thierry Delaitre¹, Gabor Kecskemeti², Stephen Winter¹

¹Centre for Parallel Computing, University of Westminster
115 New Cavendish Street, London, W1W 6UW
e-mail: gemplca-discuss@cpc.wmin.ac.uk

²MTA SZTAKI, 1111 Kende utca 13
Budapest, Hungary

Abstract

Currently several production Grids offer their resources for academic communities. Access to these Grids is free but restricted to academic users. As so, the Grids offer only basic quality of service guarantees and minimal user support. This incorporates Grid portals without workflow editing and execution capabilities, brokering with no QoS and SLA management, security solutions without privacy and trust management. Today's Grids do not provide any kind of support for running legacy code applications, and do only very basic accounting without any billing functionalities. This academic experimental phase should be followed by opening up the Grid to non-academic communities, such as business and industry. The widening of the Grid user community defines additional requirements. This paper discusses how the GEMPLCA legacy code architecture is extended in order to fulfil the requirements of commercial production Grids. The aim of our research is to facilitate the seamless integration of GEMPLCA into future commercial Grids. GEMPLCA, in this way, could serve as a reference implementation for any third party utility service that can be added to production Grids in order to improve their usability.

1. Introduction

Grid computing has the potential to move from academic and research communities towards commercial applications and become a major force in business and industry. Some companies made huge investments in compute, storage and other resources in the 1990s, which sit idle most of the time. Some other companies do not have the resources they need in order to solve their business problems. Grids offer solutions for both categories of companies by integrating their heterogeneous resources and providing on-demand computing power. Grid computing connects distributed resources of companies, harnesses their collective resources, and manages them as a single resource.


To make Grid computing available for business and industry, commercial production Grids have to be created that provide stable, production quality infrastructures based on Grid economics. These commercial production Grids should utilise the experiences of existing academic production Grids and should be built on available Grid software technologies. However, several aspects, like virtual organisation management (privacy, security and trust), performance-oriented service management

(brokers and information systems) and user-friendly problem solving environments (collaborative Grid portals) have to be added to or significantly improved in current academic Grids in order to fulfil business requirements. Today's academic production Grids are not based on Grid economics, they do not provide reliability and robustness required for business and industry applications, and they do not handle Quality of Service (QoS) and Service Level Agreements (SLA). Production Grids can efficiently serve the research community even without these attributes which, on the other hand, are crucial for a wider industrial take-up of Grid technology.


GEMPLCA [1], Grid Execution Management for Legacy Code Applications is a generic solution in order to deploy legacy applications as Grid services without modifying or even accessing the original source code. GEMPLCA has been successfully used to Grid-enable several applications [2] [3], and has been offered as a service on the UK National Grid Service (NGS) [4] and the WestFocus Grid [5]. In order to support these academic Grids at production level additional features, like dynamic account management and GEMPLCA service monitoring had to be added to the original architecture. However, in order to

This paper explores some of the enhancements that have already been added to GEM/LCA, or currently under specification and development aiming to fulfil the requirements of both academic and commercial production Grids.

There are several production Grid systems, like the TeraGrid [6] and the Open Science Grid (OSG) [7] in the US, or the EGEE Grid [8] and the UK National Grid Service [4] in Europe, that already provide reliable production quality access to computational and data resources for the academic community. All these Grids were set up as resource-oriented Grids based on Globus Toolkit version 2 (GT2) [9]. All these consider moving towards a service-oriented architecture using either gLite [10] or GT4 [11]. They also consider providing service not only for the academic communities but for business and industry too. These objectives require significant enhancement of these Grid infrastructures and also the incorporation of additional user support services.



Log in



EDITING SEQUENTIAL TRAFFIC SIMULATOR (FORK/NICHO/MACRO/TUR ON HTTP://GNS-CLUSTER.CP.WMIM.AC.UK:50084/WSRP/SERVICES/GRID-DATA.BLAC.UK AT TGS)

jobType

id

minimumProcessors

maximumJob

maximumProcessors

output

count

error

jobManager

executable

description

single ☐

parallel ☐

2

stdout

count

error

jobManager

executable


description

Publish legacy code

Save Settings

Argument setup											
name	file	order	fixed	input/output	mandatory	range	type	defaultValue	commandline	initialvalue	action
+	Yes	0	No	Input	No	null	Input road network	file	Yes	hydev7.net	Edit Remove
+	Yes	1	No	Input	No	null	Input turns file	file	Yes	hydev7.tlm	Edit Remove
+	Yes	2	No	Output	No	null	Output macroscopic trace file	generic trc	Yes	generic.trc	Edit Remove
+	Yes	3	No	Input	No	null	Number of steps	Yes	500	Edit Remove	
+	Yes	4	No	Input	Yes	null	Macroscopic trace file	Yes	macro	Edit Remove	
+	Yes	5	No	Input	Yes	null	Number of cars per lane	Yes	15	Edit Remove	
+	Yes	6	No	Input	Yes	null	time period of sim. steps	Yes	200	Edit Remove	
+	Stdout	Yes	7	No	Output	Yes	standard output file	No	stdout	Edit Remove	
+	Stderr	Yes	8	No	Output	Yes	standard error file	No	stderr	Edit Remove	
										New Argument	

[Save LCTD](#) [Reset](#)



GEMLCA provides the capability to convert legacy codes into Grid services. However, end-users still require a user-friendly Web interface (portal) to access the GEMLCA functionalities: to deploy, execute and retrieve results from legacy applications. Instead of developing a new custom Grid portal, GEMLCA was integrated with the workflow-oriented P-GRADE Grid portal [15] extending its functionalities with new portlets. The P-GRADE portal enables the graphical development of workflows consisting of various types of executable components (sequential, MPI or PVM

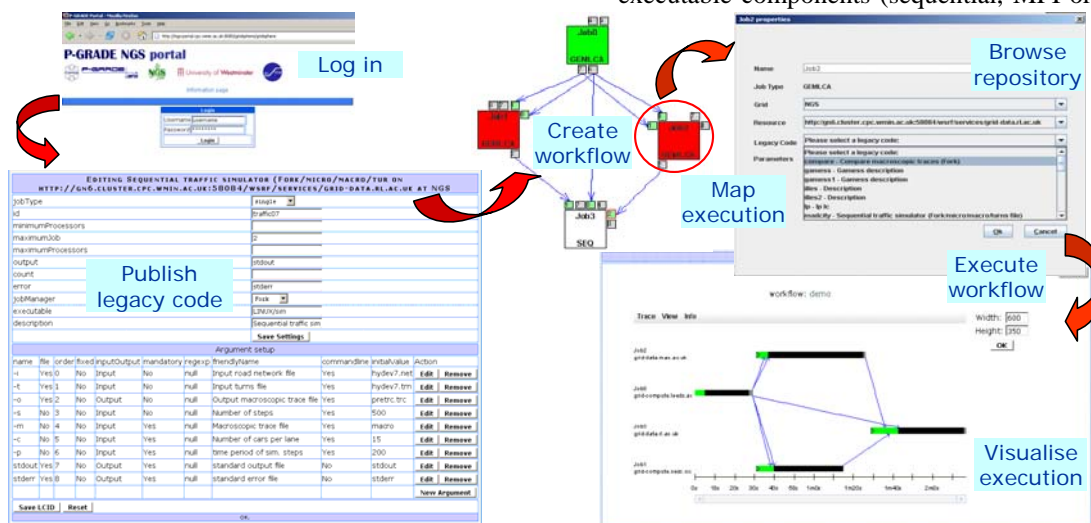


Figure 1: Functionalities of the P-GRADE/GEMLCA Portal

programs), execution of these workflows in Globus-based Grids relying on user credentials, and finally the analysis of the correctness and performance of applications by the built-in visualisation facilities. The portal is based on the GridSphere [16] portal framework and the workflow manager subsystem is currently implemented on top of Condor DAGMan [17].

Following the integration of GEMMLCA and the P-GRADE portal, end-users can easily construct workflow applications also including legacy code services running on different GEMMLCA Grid resources. The workflow manager of the portal contacts the selected GEMMLCA resources, passes them the actual parameter values of the legacy code, and then it is the task of the GEMMLCA Resource to execute the legacy code with these parameter values. The other important task of the GEMMLCA Resource is to deliver the results of the legacy code service back to the portal. The P-GRADE portal was also extended with the GEMMLCA Administration portlet. This portlet manages the XML-based Legacy Code Interface Description (LCID) files, which describe the execution environment and the parameter set of legacy applications. The portlet creates automatically the LCID files and uploads them to the appropriate directory of the GEMMLCA resource.

The functionalities of the integrated P-GRADE/GEMMLCA portal are represented on Fig. 1. It shows the GEMMLCA Administration portlet to publish a new legacy code, the workflow editor to graphically create a new workflow, define its properties and map its execution to Grid resources, and finally to visualize the execution of the workflow. For more detailed description of GEMMLCA and the P-GRADE portal please refer to [1] and [15], respectively.

In order to offer GEMMLCA as a production level service through the P-GRADE portal for the UK NGS, some further developments of the architecture were necessary: GEMMLCA is extended with dynamic user management and service monitoring capabilities. Work is also in progress to define and implement an accounting and charging service that administers GEMMLCA resource usage and provides billing information. This extension is not necessary in case of the NGS but crucial for commercial Grid applications. The next chapter describes these current research efforts in detail.

3. GEMMLCA Extensions to Support Production Grids

3.1 Dynamic User Management in GEMMLCA

As GEMMLCA uses local job managers, like Condor or PBS, to execute legacy code jobs through GT4 job submission, it requires a secure run-time environment. In order to achieve this Grid certificates have to be mapped to local user accounts. However, in case of a production Grid it is not scalable to create user accounts and do the mapping manually whenever a new user is added. In a production Grid environment users' Grid credentials have to be mapped dynamically to local user accounts in a completely user-transparent way. Current GT2-based production Grids all tackle this problem. However, there are only limited solutions for GT4-based Grids. GEMMLCA is capable to submit jobs to and work seamlessly together with GT2-based Grids like the UK NGS but its internal implementation is based on GT4. Because of this, dynamic user management of GT4-based GEMMLCA resources had to be resolved. This section describes a unique architecture how dynamic mapping can be integrated into GT4-based Grid services like GEMMLCA, allowing the seamless integration of these services into production Grids.

To provide a scalable user management GEMMLCA was integrated with the Workspace Management Service [18] using its identity mapping and dynamic account pooling features. The first feature maps Grid certificates into local accounts at run-time without manual intervention, while the second feature provides local accounts on demand. The previous GEMMLCA security implementation [19] used the grid-mapfile to check authorization of Grid service requests. To avoid manual interaction, authorisation of GEMMLCA services, such as *GLCList*, *GLCProcess* and *GLCAdmin* [1], was adapted to the WMS Authorisation and Mapping Callout to make GEMMLCA scalable. To get a workspace, *GLCProcess* issues a request on the user's behalf to the WMS to create and lease a workspace for the user. Leased workspaces allow Grid users to access GEMMLCA resources and allow them to make subsequent service requests. The *GLCProcess* can extend the lease as required, for example until the service completes, during the execution of the legacy codes using a thread associated with the *GLCProcess* environment.

The GEMMLCA lifecycle with WMS incorporates the following steps (Figure 2) (Please note that the

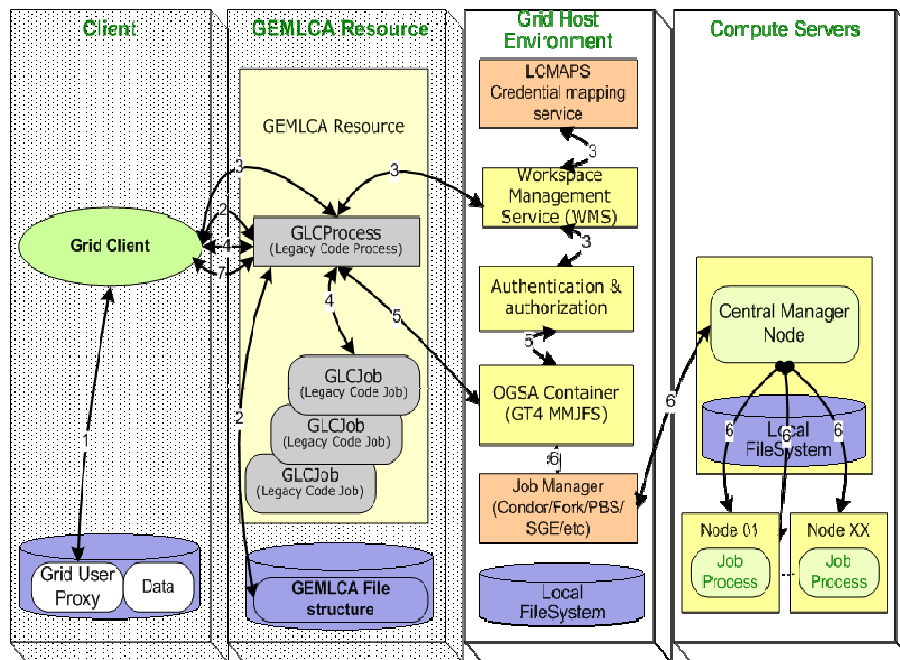


Figure 2: GEMLCA Lifecycle with Workspace Management

original GEMLCA lifecycle and its detailed description are available in [1]. Here only the changes required by dynamic account management are presented):

1. The user signs its security certificates in order to create a Grid user proxy
2. A Grid client creates a restricted Grid Legacy Code Process (*GLCProcess*) instance with no defined workspace, using the GEMLCA file structure.
3. The *GLCProcess* instance forwards the Grid user credential to the Workspace Management Service (WMS) that checks whether a workspace has been previously assigned to the user. If has not been, WMS converts global user identity to local one using the LCMAPS and selects a workspace from the workspace pool assigning it to the user with a lease before creating a *GLCProcess* environment. If the lease is about to expire *GLCProcess* contacts the WMS in order to extend it. As these steps are programmed within the *GLCProcess*, the dynamic creation and mapping of workspaces are totally transparent from the users' point of view.
4. Having a workspace allocated for the user, the Grid client sets and uploads the input parameters needed by the legacy code program exposed by the *GLCProcess*,
5. The Grid user credential is delegated by the *GLCProcess* from the client to the underlying Grid Host Environment for the allocation of resources. For example, in case of a Globus-based implementation the resource allocation is the task of the Master Managed Job Factory Service (MMJFS). MMJFS validates global user identity mapped to a workspace.
6. The Grid middleware contacts the appropriate job manager (Condor, Fork, PBS etc.) that allocates resources and executes the parallel or sequential legacy code on the Compute Servers.
7. As long as the client credentials have not expired and the *GLCProcess* is still alive, the client can contact GEMLCA for checking job status and retrieving partial or final results at any time.

3.2 Resource & Service Monitoring

In order to offer GEMLCA legacy code services for production Grid systems, automatic testing of these services is inevitable. The GEMLCA Monitoring Toolkit (GMT) was developed to provide monitoring information based on probes (scripts) checking the status of GEMLCA resources. Using the GMT, system administrators are automatically alarmed when a test fails and can also request the execution of any test on-demand. The GMT also assists P-GRADE portal users when mapping the execution of workflow components to resources by offering only verified Grid resources when creating a new workflow or when rescuing a failed one.

The GMT is based on MDS4 (Monitoring and Discovery System) that is part of the Globus distribution. MDS4 is capable to collect, store and index information about resources, respond to queries concerning the stored information using the XPath language, and control the execution of testing and information retrieval tools built as part of the GEMLCA Monitoring Toolkit. It can be extended and tailored to obtain specific information by means of polling resources, subscription to obtain notifications regarding changes to the state of specific resources, and execution of test and information collection

probes.

As part of the GMT, several probes were implemented that collect information concerning the state of basic Globus services, local job manager functionality, and GEMLCA services. The probes can immediately be used as standalone tools executed automatically from the MDS by means of an XML configuration file, or manually from a command line interface, and they are also integrated into the P-GRADE portal assisting both system administrators and end-users.

System administrators can configure the MDS4 service to run the various probes at pre-defined intervals. The results are collected by a portlet that is integrated into the P-GRADE portal. Administrators can also select a specific probe from a drop-down list displayed by a portlet and run it to verify the state of a specific service at a specific site on demand. GMT probes can also assist end-users when mapping a new workflow execution onto available Grid resources, or when rescuing and re-mapping a failed workflow. In the latest P-GRADE portal release mapping of workflow components to underlying resources is done either manually by the end-user, or by a broker, for example in EGEE by the LCG broker. The GMT aims to support manual mapping, when no broker is available, by dynamically querying the MDS4 during workflow creation time, and

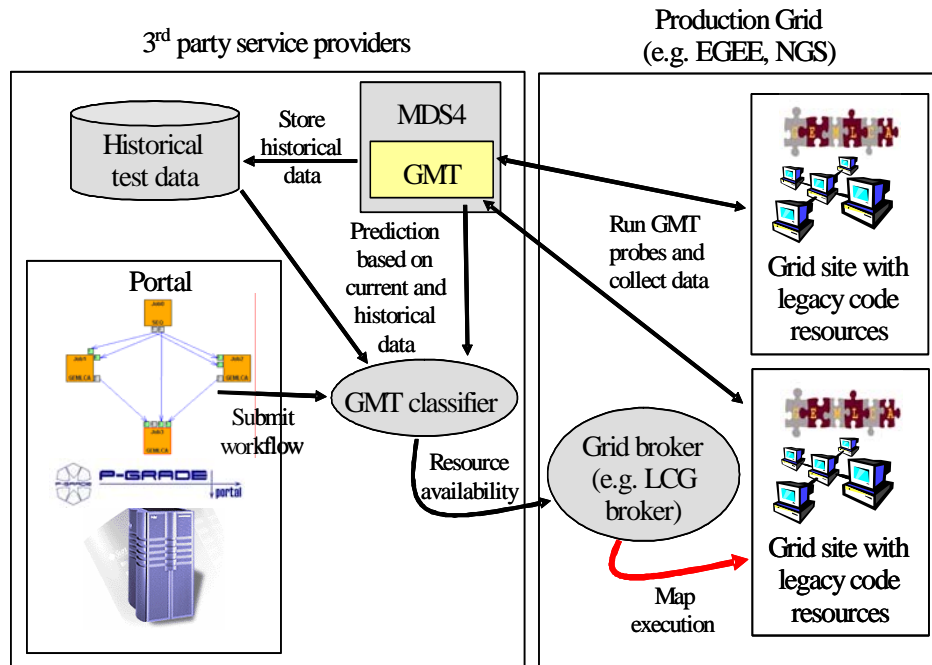


Figure 3: GMT based resource availability prediction

offering only those GEMMLCA resources for mapping those latest GMT test results were positive. Although, this does not guarantee that the resource will actually work when executing the workflow, the probability of a successful execution will be significantly increased. Work is also undergoing to connect the GMT to the LCG resource broker, as illustrated on Fig. 3. GMT, as

oriented architecture, the Grid Accounting and Charging architecture (GAC) has been elaborated for accounting and charging for usage of GEMMLCA resources and services. The architecture, given on Fig. 4, incorporates the accounting service, the charging service, the GAC database with accounts and usage records, and portlets to manage accounting and charging.

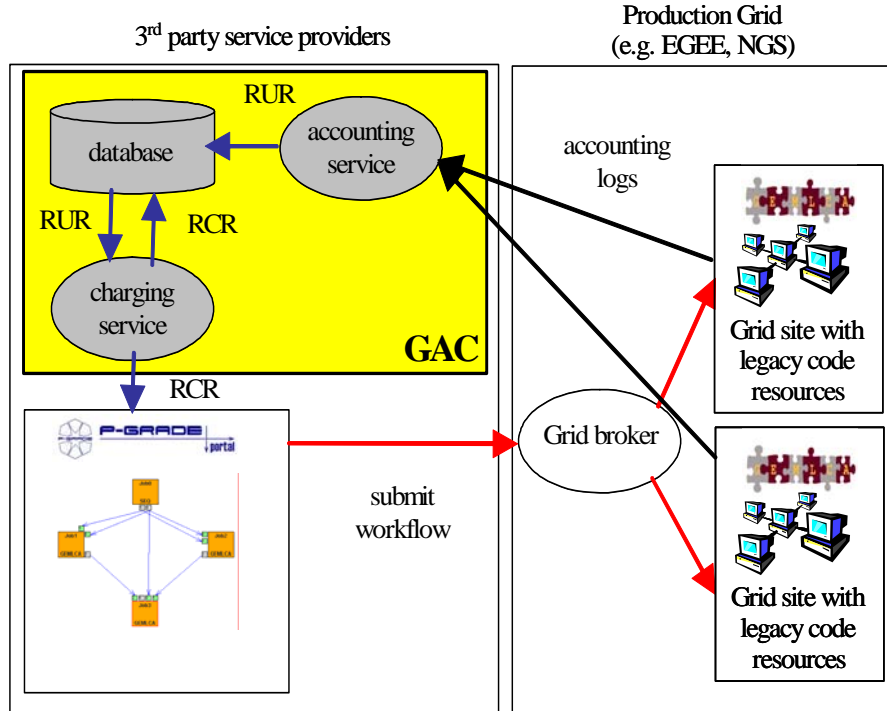


Figure 4: Grid resource usage accounting and charging

shown on the figure, runs probes on the production Grid resources and, besides updating the MDS indexing service, also creates a historical database. When the portal submits a workflow, a classifier component runs data mining algorithms on this historical data and determines which resources are “very likely to be alive”. This information can be passed to the production Grid broker, for example in case of an LCG broker within the JDL (Job Description Language) file. The broker then maps the execution to the appropriate resources taking now the GMT provided information into consideration too.

3.3 GEMMLCA Accounting And Charging

After analysing existing accounting and charging solutions, such as DGAS [20], GREAC [21], GridBank [22] and taking into consideration accounting, brokering and charging requirements, business models and usage scenarios, a service-

There are two players, who use GAC; consumers, i.e. Grid users, and providers, i.e. Grid resource and service providers. Both players have accounts in the GAC database. GAC is built on the Resource Usage Record (RUR) [23]. RUR, which is an XML document, contains the components given in Table 1.

GEMMLCA generates accounting logs that contain the job data, plus the resource and user IDs (or certificates). The resource and user data is stored in the GAC database in order to minimise the size of the accounting log. The GAC accounting service has two main tasks: first, it manages accounts, i.e. it opens, closes accounts, retrieves, displays and updates accounts’ data, and handles transactions, i.e. it deposits and withdraws money to/from accounts. Secondly, it processes the accounting log of each job submission and service request, defines the usage time, and creates

and stores the RUR in the corresponding resource and user accounts.

resource data	user data
host name / IP address	host name / IP address
certificate name	certificate name
host type	
local job ID	
wall clock time + price	
user CPU time + price	job data
system CPU time + price	
main memory + price	
secondary storage + price	
I/O channels + price	
	job ID
	application name
	job start date
	job end date

Table 1: Resource Usage Record

The charging service uses the RUR as input data to generate the Resource Charge Record (RCR), which is given in Table 2. The charging service calculates charges according to the business model of the resource and service. The business model defines how to charge for usage, how to pay, etc. The charging service should be able to support different charging policies, such as “pay before use”, “pay after use”, “pay as you go”, etc.

user	certificate name
resource	certificate name
job ID	
resource	business model
resource	usage date
resource	usage time
amount to be charged	

Table 2: Resource Charge Record

In order to charge for GEMMLCA service usage, GEMMLCA legacy code services have to be extended with economic service data. This economic service data includes attributes like pricing mechanism and currency, and additional information like liability, compensation and refund policies. As GEMMLCA describes the legacy codes in the XML-based Legacy Code Interface Description file (LCID), this file can be extended with the economic data.

The charging service stores the RCR in the GAC database and also sends the RCR to both the resource/service provider's and the user's

accounts. Having the RCR the charging service transfers the calculated amount from the user's account to the provider's account.

To provide access to the providers' and users' accounts and manage them, an account portlet will be added to the GEMMLCA portal.

4. Conclusions and Further Work

Current production Grid systems serve only academic communities and do not charge for their services. However, extending the usage of these Grids towards businesses and industry is the next inevitable step for the widespread take-up of Grid computing. This step requires several enhancements of current Grids. Our paper described some necessary extensions to the current GEMMLCA architecture in order to support these next generation Grids. User support services, like GEMMLCA, are inevitable for future Grid systems and will enable the real usability of these infrastructures. Providing these pioneering developments in GEMMLCA we are intending to create reference implementation for similar user support services.

This paper concentrated on the latest ongoing research and development work that extends the architecture with dynamic user management, resource monitoring and availability prediction, and accounting and charging capabilities. Most of these concepts are already implemented as prototypes and their full integration into the GEMMLCA architecture is currently work in progress.

References

- [1] T. Delaitre, T. Kiss, A. Goyeneche, G. Terstyanszky, S. Winter, P. Kacsuk: GEMMLCA: Running Legacy Code Applications as Grid Services, Journal of Grid Computing Vol. 3. No. 1-2. June 2005, Springer Science + Business Media B.V., Formerly Kluwer Academic Publishers B.V. ISSN: 1570-7873, pp 75-90
- [2] A. Goyeneche, T. Kiss, G. Terstyanszky, G. Kecskemeti, T. Delaitre, P. Kacsuk, S.C. Winter, Experiences with Deploying Legacy Code Applications as Grid Services using GEMMLCA, Conf. Proc. of the European Grid Conference, February 14 -16, 2005, Science Park Amsterdam, The Netherlands, Volume editors: P.M.A. Sloot, A.G. Hoekstra, T. Priol, A. Reinefeld, M. Bubak, pp 851-860, ISBN: 3-540-26918-5
- [3] A. Tarczyski, T. Kiss, D. Qu, G. Terstyanszky, T. Delaitre, S. Winter, Application of Grid Computing for Designing a Class of Optimal

- Periodic Nonuniform Sampling Sequences, Conf. Proc. of the Grid-Enabling Legacy Applications and Supporting End Users Workshop, within the framework of the 15th IEEE International Symposium on High Performance Distributed Computing , HPDC'15, Paris, France, June 19-23, 2006
- [4] The UK National Grid Service Website, <http://www.ngs.ac.uk/>
 - [5] The WestFocus Grid Alliance Website, <http://www.gridalliance.co.uk/>
 - [6] The TeraGrid Website, <http://www.teragrid.org/>
 - [7] The Open Science Grid Website, <http://www.opensciencegrid.org/>
 - [8] The EGEE Website, <http://public.eu-egee.org/>
 - [9] The Globus Toolkit GT2, <http://www.globus.org/>
 - [10] EGEE gLite version 1.5 Documentation, <http://glite.web.cern.ch/glite/documentation/default.asp>
 - [11] The Globus Toolkit GT4, <http://www.globus.org/>
 - [12] Y. Huang, I. Taylor, D. W. Walker, "Wrapping Legacy Codes for Grid-Based Applications", Proceedings of the 17th International Parallel and Distributed Processing Symposium, workshop on Java for HPC), 22-26 April 2003, Nice, France. ISBN 0-7695-1926-1
 - [13] B. Balis, M. Bubak, M. Wegiel. A Solution for Adapting Legacy Code as Web Services. In Proceedings of Workshop on Component Models and Systems for Grid Applications. 18th Annual ACM International Conference on Supercomputing, Saint-Malo, July 2004
 - [14] D. Gannon, S. Krishnan, A. Slominski, G. Kandaswamy, L. Fang, "Building Applications from a Web Service based Component Architecture, in "Component Models and Systems for Grid Applications" edited by V. Getov and T. Kiellmann, Springer, 2005, pp 3-17, ISBN 0-387-23351-2.
 - [15] P. Kacsuk G. Sipos: Multi-Grid, Multi-User Workflows in the P-GRADE Grid Portal, Journal of Grid Computing, , Springer Science + Business Media B.V., Vol. 3 No. 3-4 Dec, 2005, pp 221-238, ISSN: 1570-7873
 - [16] J. Novotny, M. Russell, O. Wehrens: GridSphere, "An Advanced Portal Framework", Conf. Proc. of the 30th EUROMICRO Conference, August 31st - September 3rd 2004, Rennes, France.
 - [17] James Frey, Condor DAGMan: Handling Inter-Job Dependencies, <http://www.bo.infn.it/calcolo/condor/dagman/>
 - [18] Globus Workspace Management Service, <http://www.globus.org/>
 - [19] G.Terstyansky, T. Delaitre, A. Goyeneche, T. Kiss, K. Sajadah, S.C. Winter, P.Kacsuk, Security Mechanisms for Legacy Code Applications in GT3 Environment, Conf. Proc. of the 13th Euromicro Conference on Parallel, Distributed and Network-based Processing, Lugano, Switzerland, February 9-11, 2005
 - [20] Andrea Guarise: Grid Accounting in EGEE using DGAS. Current practices Terena Networking Conference, Poznan, June 8, 2005
 - [21] Shawn Mullen, IBM, Matt Crawford, FNAL, Markus Lorch, VT, Dane Skow, FNAL – "Site Requirements for Grid Authentication, Authorization and Accounting" GFD-I.032 – GGF, October 13, 2004
 - [22] Alexander Barmouta, Rajkumar Buyya: GridBank: A Grid Accounting Services Architecture (GASA) for Distributed System Sharing and Integration
 - [23] RUR, Resource Usage Record Working Group, Global Grid Forum, http://www.gridforum.org/3_SRM/ur.htm