HAL
open science

How To Use GRID Technology for Building Next Generation Learning Environments

Nicola Capuano, Francesco Orciuoli, Angelo Gaeta, Giuseppe Laria, Pierluigi Ritrovato

To cite this version:

Nicola Capuano, Francesco Orciuoli, Angelo Gaeta, Giuseppe Laria, Pierluigi Ritrovato. How To Use GRID Technology for Building Next Generation Learning Environments. Proceedings of the 2nd International LeGE-WG Workshop "e-Learning and Grid Technologies: a fundamental challenge for Europe"., 2003, Paris, France. pp.8. hal-00190093

HAL Id: hal-00190093

https://telearn.hal.science/hal-00190093

Submitted on 23 Nov 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

How To Use GRID Technology for Building the Next Generation Learning Environments

Nicola Capuano, Angelo Gaeta, Giuseppe Laria, Francesco Orciuoli, Pierluigi Ritrovato
CRMPA Centro di Ricerca in Matematica Pura ed Applicata
C/O DIIMA – Università degli Studi di Salerno
Via Ponte Don Melillo – 84084 Fisciano (SA), Italia
Dipartimento di Ingegneria dell'Informazione e Matematica Applicata
Università degli Studi di Salerno
Via Ponte Don Melillo – 84084 Fisciano (SA), Italia
e-mail: {capuano, agaeta, laria, orciuoli, ritrovato}@crmpa.unisa.it

Abstract

Grid technologies promise to improve the way we think about e-learning allowing wide-scale learning resources sharing in heterogeneous and geographically distributed environments consenting, in this way, the implementation of distributed learning spaces where different organizations and individuals are able to cooperate in pursuing similar and complementary learning and training objectives. But is the e-learning ready for this evolution? In this paper we try, starting from an existing e-learning platform named IWT, to sketch a possible migration path toward a Grid based environment. IWT was selected because it presents a flexible, service-oriented, layered architecture suitable for migration in an OGSA compliant environment. The new approach will provide more flexibility, in fact, it will possible to leverage on the resources distributed across the Grid in order to build the learning experience that best fit student requirements. A use case scenario is also provided in order to emphasize differences between the two approaches. After a brief introduction of Grid and Learning technologies (section 1), a brief summary of IWT is given and the execution scenario is shown (section 2). A possible OGSA-based deployment of IWT is then sketched together with the new version of the execution scenario (section 3). Some conclusions are, then, presented (section 4).

1. INTRODUCTION

Grid and learning technologies were born to solve different issues in different domains. The first, thought as the evolution of metacomputing, address issues related to access provisioning, coordinated resource sharing and problem solving in dynamic, multi institutional virtual organization [1]. This "sharing capability" is highly controlled, with resource providers and resource consumers defining what is shared, who is allowed to share and what are the conditions under whose sharing is allowed, based on resource management and security policies.

Learning technologies instead are tied to the binomial instruction/learning and how it is changed due to the explosive increase of the web and the web technologies. E-Learning refers to learning that is delivered or enabled via electronic technologies such as internet, television, videotape, intelligent tutoring system and computer-based training. This model of instruction/learning has many advantages with respect to classical models:

- a better interaction between the learner and the learning resources he uses i.e. the learning is not passive;
- learning can happen anytime and anyplace i.e. there are not boundaries tied to time and place;
- a tutor, or the learner himself, is able to monitor the progress and to customize the learning experience basing on learner skills and preferences.

But actually there are some drawbacks related to current learning solutions. First of all they are mainly focused on the content delivery, leaving in the background the collaborative view. This is also due to the implementation of distance learning platform themselves, in which existing learning objects are platform-dependent and cannot be used outside the system. This makes more complicated the collaboration between actors of different systems. Furthermore, current learning platforms only support a specific learning-domain and are not able to support learning in different domains.

Here, but not limited to these issues, is where Grid technologies can help. Through the adoption of these technologies it is possible a wide-scale learning resource sharing in heterogeneous and geographically distributed environments, the implementation of learning organizations in which different actors (universities, teachers, learners), sharing a common target, are able to cooperate to obtain a result.

Grid technologies are now moving towards a service oriented view by the definition of the Open Grid Service Architecture (OGSA) [2] that, joining Web Service and Grid technologies, defines an open and extensible framework for distributed and highly collaborative applications. This is done through the definition of Grid Services, an extension to standard Web Services able to manage lifetime and to permit introspection of the service itself [3]. This could be the missing link in order to obtain an interactive, open and collaborative environment for distance learning build upon emerging network technologies and upcoming standards, open to the integration of third party solutions and/or services.

This paper presents a possible architecture suitable for a learning grid obtained as re-engineering of an already existing traditional e-learning platform named IWT.

2. THE INTELLIGENT WEB TEACHER PLATFORM

Intelligent Web Teacher (IWT) is a distance learning platform realized by CRMPA in order to fill up the lack of support for flexibility and extensibility in existing e-learning systems. IWT provides flexibility and extensibility characteristics for contents and services at a low level and for strategies and models at an higher level. Furthermore, IWT platform provides user-tailored didactic experiences based on user preferences and competences in order to offer personalized learning.

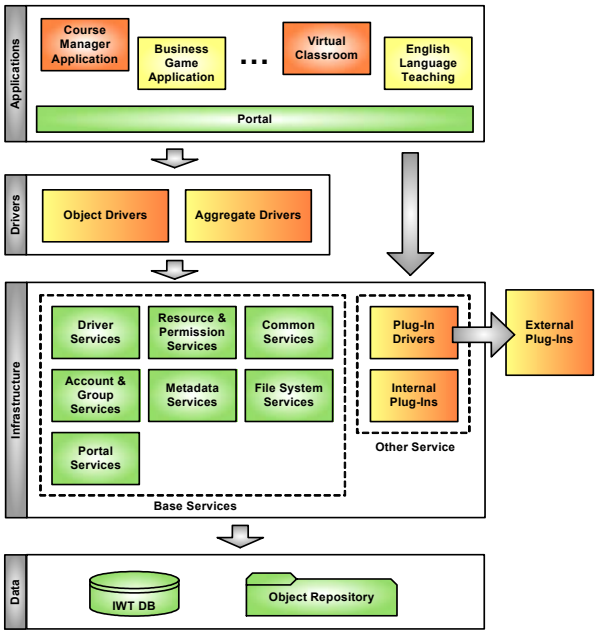

FIGURE 1. IWT Logical Architecture.

IWT arises from the consideration that every didactic/formative context requires its own specific e-learning solution. It is not thinkable to use the same application for teaching, for instance, foreign languages at primary schools and mathematical analysis at universities. It should be not only the content to vary but also its didactic model, the typology of the formative modules to be used, the application layout and, also, all the tools connected. In practice, the need to introduce the e-learning in a new didactic/formative context brings to hard work for analysts, engineers and programmers. IWT solves this problem with a really innovative solution (at the level of technologies and of methodologies) modular and extensible so to become the foundation for building up a virtually infinite set of applications for either traditional or

innovative e-learning.

In the next paragraph, we want to go deeper in the details of IWT platform showing its logical architecture.

2.1 IWT Architecture

IWT logical architecture is divided in four main layers as presented in figure 1.

The first layer at the bottom of the stack is the ***Data Layer*** that provides a way to store persistent objects as learning resources, user account information, resource indexes, etc. This layer is composed by two storage mechanisms, the first, named *Object Repository*, provides to each accounted user a space for storing information and data (i.e. a Web file system). Single user can create folders and upload or download files in their folders, can create learning resources (formed by a content and a metadata [10]) directly in his/her piece of repository. The second storage mechanism, named *IWT DB*, is a relational database, that maintains data related to user account, user groups and indexing structures (metadata) used for efficiently retrieve learning material stored in the *Object Repository*.

The second layer of the stack is the ***Infrastructure Layer*** that provides *Base Services* towards higher layers and provides also the capability to extend the services set with *Other Services*. *Base Services* are exposed as API (Application Program Interface) instead, we can add new services installing some modules called plug-ins and plug-in drivers. Typical *Base Services* provided by the Infrastructure Layer include, but they are not limited to*, Account & Group Services*; *File System Services* providing an high level view of the *Object Repository*; *Resource & Permission Services* handling permissions over objects stored in the IWT data layer; *Knowledge & Metadata Services* providing a metadata-based searching and indexing engine for learning material in *Object Repository*; *Driver Services* providing a set of functionalities for driver handling.

The *Other Services* in IWT are represented by plug-ins. A Plug-In is an additional module installed in IWT platform in order to provide new services (domain independent or domain specific) to higher layer of IWT architecture. There are *internal plug-ins* and *external plug-ins*, the latter need a standard interface called *plug-in driver* for their integration with IWT. As examples of plug-ins we want to mention:

- *Student Model Plug-In* (internal plug-in) is a module that allows learner profile handling;
- *Ontology Plug-In* (internal plug-in) is a module that allows to structure knowledge about a specific domain using ontologies [5] [7];
- *LIA* (Learner Intelligent Advisor, external plug-in) provides customisation of didactic experiences based on target objectives given by teacher, knowledge model chosen by teacher and student preferences and competences; manages student models [6] [9].

The third layer of the stack is the ***Driver Layer***. *Drivers* are pluggable components used to extend IWT services and IWT content types. There are three types of *Drivers*:

- *Object Drivers*, each of them manages, in a transparent way, a specific type of content (examples of contents are lesson pages, multiple choice tests, etc.);
- *Aggregate Drivers*, that manage complex objects (simulations, virtual experiments, etc.) arising from the aggregation of simpler objects.
- *Plug-In Drivers*, are required to adapt for IWT the work of external Plug-Ins.

A *Driver* realizes IWT extensibility, in fact, *Object* and *Aggregate Drivers* can dynamically extend the set of IWT managed contents, instead *Plug-In Drivers* extend the set of IWT managed services.

The highest layer in the stack is ***Application Layer*** in which we find the specific applications we want to realize on the IWT platform.

As said before, IWT has an highly modular platform. Due to its characteristics of flexibility and extensibility, it is fit for migrating towards a grid environment and, in our vision, it could represent the core middleware enabling the creation of the learning Grid. Before to describe how to modify the IWT platform in order to make it GRID-aware we want to illustrate a typical IWT scenario.

2.2 Delivery of a Didactic Course Scenario

In this paragraph we want to describe and analyse the main execution flow of a typical application developed using the IWT environment, namely the "Delivery of a Didactic Course".

For a better understanding of the scenario it is fundamental the introduction of the following *key abstractions*:

- **Concept** is a significant property in a particular domain i.e. "limit" is a concept in the domain of "mathematical analysis";
- **Ontology** represents a way to structure knowledge of a specific domain, they can be thought as graphs in which nodes are concepts and arcs are relations between concepts;
- **Target Concept** is an objective fixed by a teacher for a didactic experience;
- **Learning Object** is a minimal unit of didactic material, the rendering of one or more learning objects represent a didactic experience, a learning object has to be described by a metadata;
- **Metadata** is a standard set of data used in order to describe in a consistent way a resource, metadata links learning objects to concepts covered by them;
- **Student Model** is a profile composed by account information, learner preferences and competences;
- **Didactic Course Specification** is the set of specifications provided by a teacher to assemble a didactic experience, a reference to an ontology and a set of target concepts to assemble a course are needed;
- **Didactic Course Presentation** is the list of learning objects that realize the didactic course described in a Didactic Course Specification.

Now, we can move to the description of the execution flow and software modules involved in the "Delivery of a Didactic Course". Student Bob decides to access a didactic course, so he interacts with *Course Manager Application* to obtain a didactic course catalog and selects the course **A**. *Course Manager Application* asks for an instance of a suitable driver to manage the selected course **A** (see figure 2).

The *Course Manager Application* can ask to the obtained course driver instance for the delivery of course **A**. The course driver instance asks the *Driver Services* to get an instance of a *LIA Plug-In Driver* (i.e. the wrapper used to manage the ITS engine called *LIA*). *LIA* computation needs services provided by *Ontology Plug-In* and *Student Model Plug-In*. (see figure 3 and 4).

When *LIA Plug-In Driver* obtains the result of needed computation (i.e. the transformation from Didactic Course Specification into Didactic Course Presentation) it sends this result to the

course driver instance. The delivery of the course **A** presentation is realized by collaboration between the *course driver instance* and the *Course Application Manager* (see figure 5).

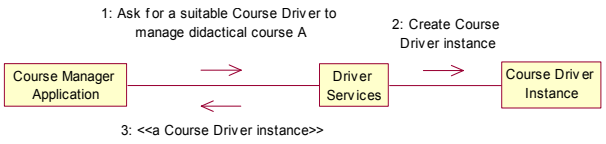

FIGURE 2. Step 1.

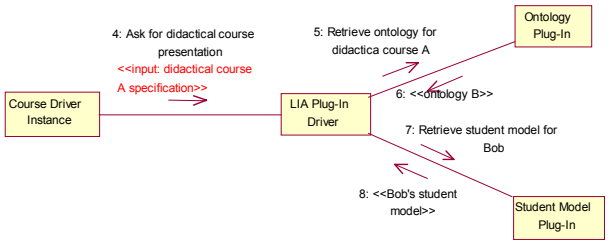

FIGURE 3. Step 2.

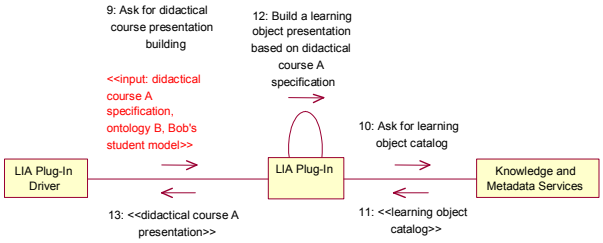

FIGURE 4. Step 3.

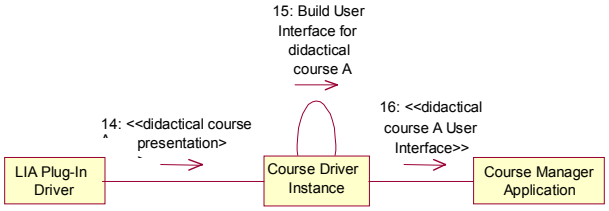

FIGURE 5. Step 4.

3. PROPOSED ARCHITECTURE ENABLING AN E-LEARNING GRID

In deploying the IWT platform in Grid environment it must be taken into account the new environment in which this platform will be developed and will operate. The new platform will be OGSA compliant, so it will inherit all the features of such architecture. Two aspects, in particular, are important:

- the openness of the architecture, where open means extensibility, vendor neutrality, and commitment to a community standardization process;
- the service orientation and virtualization, where the first is related to definition of service interfaces and the identification of protocols that can be used to invoke a particular interface, and the second is related to the encapsulation behind a common interface of diverse implementation, so everything (Resources, Learning Objects and so on) in this environment is a service.

This is done by the definition of a Grid Service [2], which is the building block of OGSA. In the figure 6 the architecture of Grid based IWT (GrIWT) is shown.

The platform is built upon a ***grid-enabling layer***. In our solution this layer should be based on existing OGSA compliant middleware such us GT3. It is composed by a *core* layer, which implements the interfaces and behaviours described in [3] (e.g. Registration, Factory, Notification) and a *base and collective* layer that implements services for Resource Management, Information Services and Data Transfer in the Grid. It enables single sign-on and security mechanism based on the Grid Security Infrastructure (GSI).

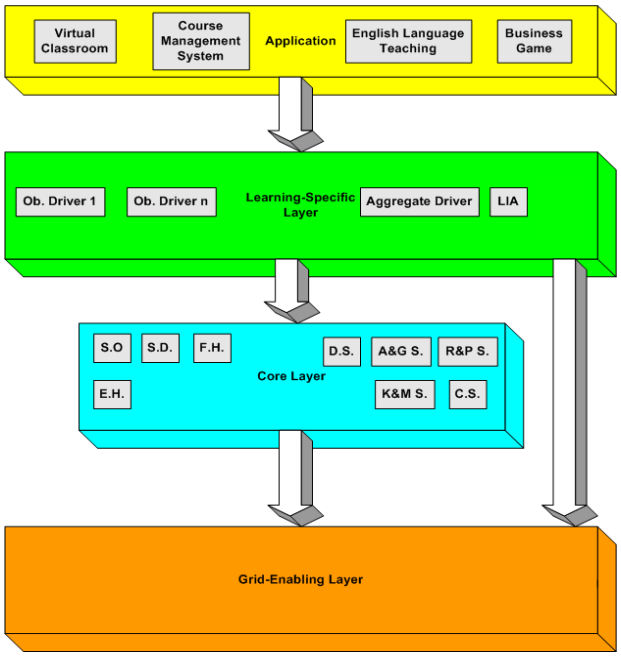

FIGURE 6. GrIWT Architecture.

The next layer represent a ***core layer*** that provides the Base Services of IWT platform with the addition of a set of services useful for the management of the architecture in a service oriented and highly distributed environment, like *Service Orchestrator* (S.O.) for the coordination of the sub-services, *Service Discovery* (S.D.) for locating and retrieving service description documents, *Failure Handling* (F.H.) to manage the various forms of failure and exceptions that can arise in this environment and *Event Handler* (E.H.) to manage events and notifications between services. These services should be OGSA compliant.

The features of flexibility and extensibility of the IWT platform are a good starting point towards the deployment of the platform in a Grid environment, but much work must be done. The services of IWT should be wrapped by GSs and extended in order to adapt them to the new environment. For example, the *Account & Group* services of IWT should provide user authentication no more in a stand-alone platform, but in a dynamic and distributed Learning Organization, providing single sign-on mechanisms. Analogously, IWT *Resource & Permission* services should provide authorization on resources distributed across the Grid and belonging to the same Learning Organization, based on access policy of the community [4].

In GrIWT there is no need for the Plug-In mechanism of IWT. In the new environment new services can be added and removed dynamically and, furthermore, external services discovered by the use of S.D., can be provided by third parties, distributed across the grid, taking the role of Content/Service Providers and can be orchestrated with other services of the platform to build sophisticated new services.

This two layers comprise the minimal set of services that the platform must have. With this set of services a Learning Management System (LMS) is able to access the Grid in a secure way, to discover services and resources and to orchestrate them with its services or resources.

On top of them, there is a ***learning-specific*** layer, including domain specific services. These services comprise Object Drivers and the Aggregate Driver as well as a set of specific services like LIA. These services should be OGSA compliant, but they could be Web Services invoked by GS, too. Like for the core layer, there is no need for the plug-in drivers. Finally, there is the ***application*** layer in which the applications can be composition of GS orchestrated between them.

In the next paragraph, we explain how the scenario, described in 2.1, changes according with the new Grid based architecture.

3.1 Delivery of a Didactic Course

The first step of the Didactic Course delivery concerns the selection of a suitable *Course Driver* for the course chosen by the student.

In this case, when the *Driver Services* is invoked by the *Course Manager Application* (step 2 of paragraph 2.1), it queries the S.D. grid service in order to obtain an appropriate reference (e.g. GSH, see [2] for details) to a Factory of the Driver hosted somewhere on the Grid. After the invocation of the Factory, the *Driver Services* receive a reference to a new instance of the *Course Driver* and returns it to the Application. Now Application can invoke the *Course Driver*, which provides the actual delivery of the course.

In this phase (step 2, 3, 4 of paragraph 2.1), the main difference, with respect to the IWT platform, is the introduction of the Orchestration service and the elimination of the *Plug-In Drivers* that become unnecessary with the migration towards a Grid Services based environment.

In order to deliver the course, the *Course Driver* (in this scenario it should be a Grid Service), invokes the *Orchestrator*, passing to it a description (defined in a specified language) of the workflow among the involved services. The *Orchestrator* runs the workflow definition and manages the interactions among services.

The Figure 7 shows how the *Orchestrator* enters in the scenario depicted in 2.1. The numeration of the lines reflects the steps described in 2.1. The dotted lines describe the logical execution flow among the services (it is the same we described in 2.1). While the continuous lines represent the actual execution flow. The *Orchestrator* manages the interactions among the Services. It is a general service, in fact, it, simply, runs the workflow description, which is generated by the opportune Driver (in this case the Driver suitable for course A).

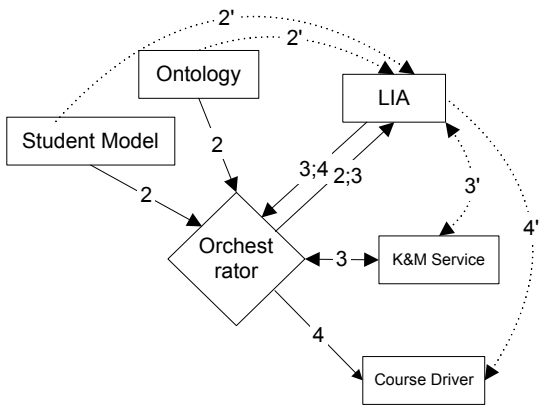

FIGURE 7. Workflow for Course Delivery.

We would like underlining the flexibility of this architecture, and the introduction of the modification to the *K&M Service* that is able to retrieve references to Learning Objects distributed around the Grid, querying the Service Discovery. This opportunity allows *LIA* to assemble more suitable courses to student profiles.

4. CONCLUSIONS

In this paper we have presented the possibility to use the Grid technologies in order to implement an innovative platform for distance and collaborative learning. We have placed particular emphasis on showing the advantages that can be obtained in deploying IWT in an OGSA compliant environment. Much work must be done both in the identification and in the implementation of the functionalities that the new platform must have but, as we argued, GrIWT can potentially facilitate the emergence of a new learning models, user centred and not content centred, built upon Grid technologies.

REFERENCES

[1] Foster, C. Kesselman, J. and Tuecke, S. *The Anatomy of the Grid.*

[2] Foster, C. Kesselman, J.. Tuecke, Nick and S. (2002) *The physiology of the Grid* .

[3] Foster, C. Kesselman, J., Tuecke, S. et al. (2002) *Grid Service Specification* draft 05.

[4] Foster, C. Kesselman, J Tuecke, S et al. (2002) *A Community Authorization Service for Group Collaboration*.

[5] Fensel, D. (2001) Ontologies: *a Silver Bullet for Knowledge Management and Electronic Commerce*. Springer.

[6] Capuano, N, Marsella. M, Salerno, S. (2000) ABITS: An Agent Based Intelligent Tutoring System for Distance Learning. *Proceedings of the International Workshop on Adaptive and Intelligent Web-Based Education Systems*. ITS 2000, Montreal, Canada.

[7] Decker, S., Fensel, D., van Harmelen, F., Horrocks, I., Melnik, S., Klein, M., and Broekstra, J. (DL2000) Knowledge Representation on the Web In: *Proceedings of the International Workshop on Description Logics*.

[8] IWT: Intelligent Web Teacher. (2002) White Paper. CRMPA.

[9] Capuano, N., Gaeta, M., Micarelli, A., Sangineto E. (2002) An Integrated Architecture for Automatic Course Generation. *Proceedings of the IEEE International Conference on Advanced Learning Technologies.* ICALT 2002, Kazan, Russia.

[10] IMS Learning Resource Meta-data Specification Version 1.2.2. (2001) Public Draft Specification. IMS Global Learning Consortium. http://www.imsproject.org/metadata/index.cfm.