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

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, allowing, 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 could 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.

Keywords: Grid Technologies, e-learning platforms, Grid aware applications, distributed learning management systems.

Introduction

Grid and learning technologies were born to solve various issues in different domains. The first, thought as the evolution of metacomputing, addresses some issues related to access provisioning, coordinated resource sharing and problem solving, towards 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 allowing the sharing, based on resource management and security policies.

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

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

Nevertheless, 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 platforms 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 limitedly to these issues, is where Grid technologies can help. Through the adoption of these technologies, we can have 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 achieve 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 the Web Service and the 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 built 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 accomplished by the re-engineering of an already existing traditional e-learning platform named IWT.

1. The Intelligent Web Teacher platform

The 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 features for low level contents and services and for higher level strategies and models. Furthermore, the IWT platform provides user-tailored didactic experiences based on the user's preferences and competences in order to offer personalized learning.

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 connected tools. In practice, the need to introduce the e-learning in a new didactic/formative context brings to harder work for analysts, engineers and programmers. IWT solves this problem with a really innovative solution (both at the technologies and methodologies level) 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 into the details of the IWT platform, showing its logical architecture.

IWT Architecture

The IWT logical architecture is divided into 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 of two storage mechanisms, the first, named *Object Repository*, provides each accounted user a space for storing information and data (i.e. a Web file system). Single users can create folders and upload or download files in their folders, can create learning resources (formed by a content and a metadata [4]) 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 also the capability to extend the services set with *Other Services*. The *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.

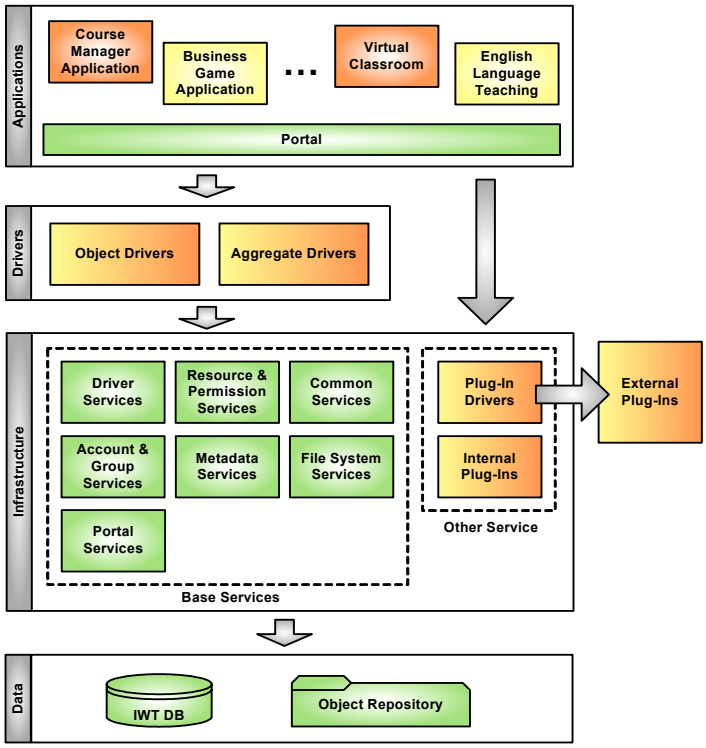

Figure 6 – IWT Logical Architecture.

The *Other Services* in IWT are represented by plug-ins. A Plug-In is an additional module installed in the IWT platform in order to provide new services (domain independent or domain specific) to the IWT architecture higher layer. 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 the following worth mentioning:

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

A *Driver* realizes the 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 the ***Application Layer*** in which we find the specific applications we want to realize on the IWT platform.

As said before, IWT has a highly modular platform. Due to its features 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 describing how to modify the IWT platform to make it Grid-aware, we want to illustrate a typical IWT scenario.

Delivery of a Didactic Course Scenario

In this paragraph we 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 this scenario, of fundamental importance is 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 a specific domain knowledge, it 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 required to assembly a didactic experience and is composed of a reference to ontology and a set of target concepts.
- Didactic Course Presentation is the list of learning objects that realize the didactic course described in a Didactic Course Specification.

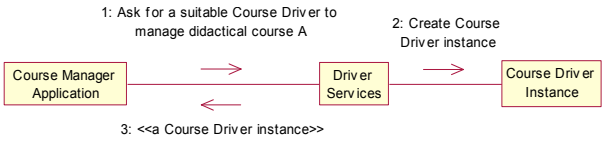

Figure 2 – Step 1.

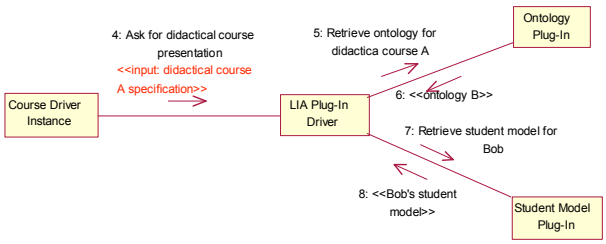

Figure 3 – Step 2.

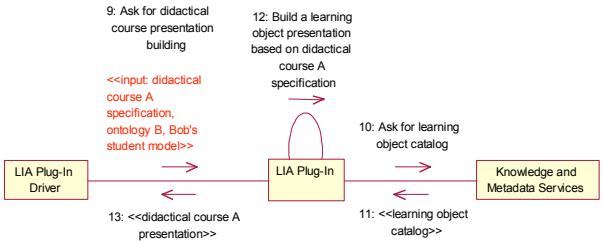

Figure 4 – Step 3.

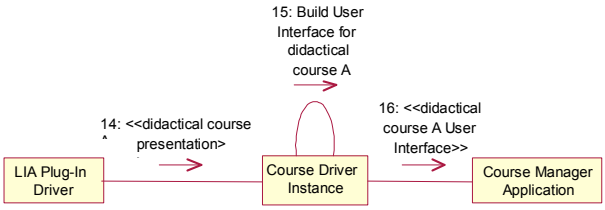

Figure 5 – Step 4.

Moving to the description of the execution flow and software modules involved in the "Delivery of a Didactic Course", we can assume that Student Bob decides to access a didactic course, so he interacts with *Course Manager Application* to obtain a didactic course catalogue and decides to select 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 the 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 the 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 the collaboration between the *course driver instance* and the *Course Application Manager* (see figure 5).

2. Proposed architecture enabling an e-Learning Grid

In deploying the IWT platform into the 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 the 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 as GT3. It is composed of 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).

The next layer represents a ***core layer*** that provides the Base Services of the 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; *Event Handler* (E.H.) to manage events and notifications between services. These services should be OGSA compliant.

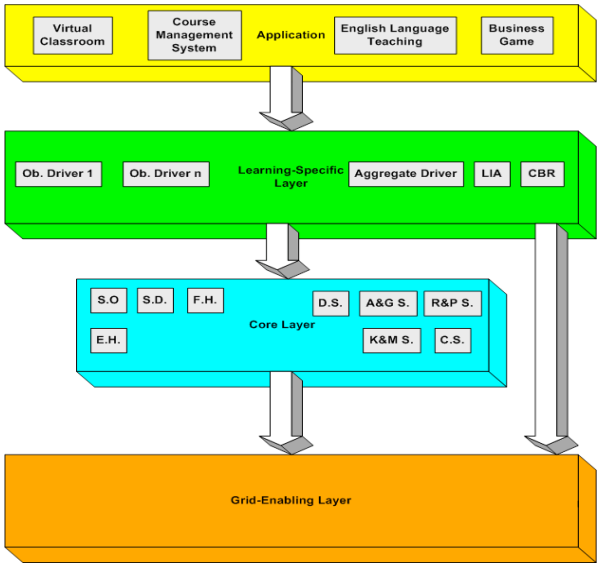

Figure 6 – GrIWT Architecture.

The IWT platform flexibility and extensibility features are a good starting point towards the deployment of the platform in a Grid environment, even if much work must be still done. The IWT services 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, the IWT *Resource & Permission* services should provide the authorization on the resources distributed across the Grid and belonging to the same Learning Organization, based on the community access policy [4].

In GrIWT the IWT Plug-In mechanism is not required. 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 new sophisticated services.

These 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 the core layer, there is no need for the plug-in drivers. At the end, there is the ***application*** layer in which the applications can be a composition of GS orchestrated between them.

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

3. Delivery of a Didactic Course

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

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, the 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*, transferring to it a description (defined in a specified language) of the workflow between the involved services. The *Orchestrator* runs the workflow definition and manages the interactions between services.

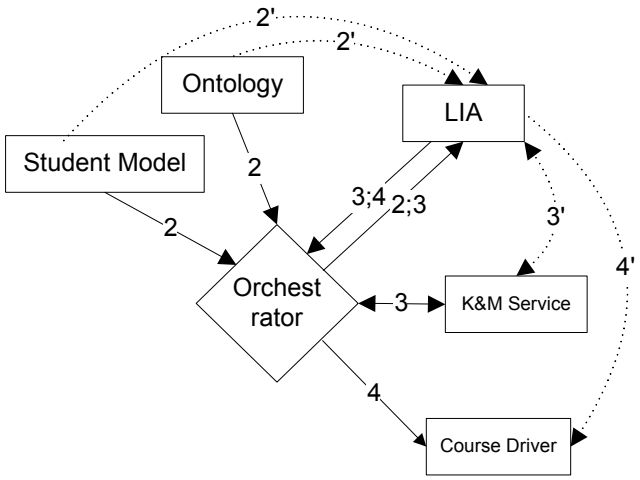

Figure 7 – Workflow for Course Delivery.

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 between 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 between the Services. It is a general service, since it simply runs the workflow description, which is generated by the appropriate Driver (in this case the Driver suitable for course A).

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

4. Conclusions

In this paper we have represented the possibility to use the Grid technologies in order to implement an innovative platform for the distance and collaborative learning. Particular emphasis has been laid on showing the advantages that can derive from deploying the IWT in an OGSA compliant environment. Much work must be done both in the identification and in the implementation of the functionalities peculiar to the new platform. As we have argued, GrIWT can potentially facilitate the birth of new learning models, which are user centred and not content centred, and built upon the Grid technologies.

References

[1] Foster, C. Kesselman and S. Tuecke, "The Anatomy of the Grid"
[2] Foster, C. Kesselman, J. Nick and S. Tuecke, "The physiology of the Grid" 2002
[3] Foster, C.Kesselman, S. Tuecke et al. "Grid Service Specification" draft 05 2002
[4] Foster Kesselmam Tuecke et al. A Community Authorization Service for Group Collaboration, 2002
[5] D. Fensel. Ontologies: a Silver Bullet for Knowledge Management and Electronic Commerce. Springer, 2001.
[6] N. Capuano, M. Marsella, S. Salerno. 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, 2000.
[7] S. Decker, D. Fensel, F. van Harmelen, I. Horrocks, S. Melnik, M. Klein and J. Broekstra: Knowledge Representation on the Web In: Proceedings of the International Workshop on Description Logics (DL2000).
[8] IWT: Intelligent Web Teacher. White Paper. CRMPA. 2002.
[9] N. Capuano, M. Gaeta, A. Micarelli, E. Sangineto. An Integrated Architecture for Automatic Course Generation. Proceedings of the IEEE International Conference on Advanced Learning Technologies. ICALT 2002, Kazan, Russia, 2002.
[10] IMS Learning Resource Meta-data Specification Version 1.2.2. Public Draft Specification. IMS Global Learning Consortium, 2001. http://www.imsproject.org/metadata/index.cfm