Learning GRID

A newsletter form the Kaleidoscope Learning GRID SIG **Chief Editor: Thanasis Daradoumis**

Newsletter #12: June 2007

Deliverable D 12.2.13.2

Prepared for the European Commission under Contract No. NoE IST-507838 as a deliverable from

Task: 12.2: Networking and Dissemination
Date of issue: 30/06/2007
Version: 1.0
Distribution: public

In this Issue:

Editorial

Welcome to the twelfth issue of the Kaleidoscope Learning GRID SIG newsletter.

The main purpose of this newsletter is to resume some of the results of recent research on resources and services deployment, services description, discovery and composition in distributed and learning grid environments.

To this end, we present an article from the Distributed, Parallel and Collaborative Systems (DPCS) research group of the Open University of Catalonia. Then we provide a review of this subject in the Technology Watch section. We expect this generates an interesting feedback and a beneficial interchange of ideas and experiences on this issue's topic.

The article presents the design of a system which allows the deployment of services in a small-sized group of computers distributed through the Internet. These groups are formed by users with a common interest, who voluntarily yield their own resources for the collaborative activities of the group. Having enough resources contributed by the members of the group, the system guarantees service availability and the fact that the deployment and execution of the services is carried out using only the resources of the group. This management is done in a completely decentralized manner, and the system is self-organized in the presence of component connections, disconnections and failures.

The research work described in this article forms part of a bigger project, called Grid4All, which we present next. Indeed, Grid4All is an EC project that embraces the vision of a "democratic" Grid as a ubiquitous utility whereby domestic users, small organizations and enterprises may draw on resources on the Internet without having to individually invest and manage computing and IT resources. This project funded by the 6° Framework Programme of the European Commission involves institutional and industrial partners.

Furthermore, the Technology Watch section shows an overview of the different languages, approaches and mechanisms that can function between discovery and matching of Grid Learning Services. In particular, we present a way to take advantage of semantic capabilities in order to achieve an efficient description, discovery, matching and composition of Grid Learning Services.

The News section reports on an effort to deploy an Iberian Grid Infrastructure, named IBERGRID, in order to better serve the computing needs of the Iberian Research community, as well as on two new promising projects, an Ibero-American project, CyTEDGrid, to provide technology for regional development, and a project that aims at the design, implementation, evaluation and distribution of an open source Grid operating system, called XtreemeOS.

Finally, When-What-Where section notifies the call for papers of the main workshops and conferences that will be held in this last term of the year.

Enjoy your read.

Thanasis Daradoumis
Learning Grid SIG Steering Committee Member

Deploying services in a decentralized collaborative environment

Featured Article by Daniel Lázaro, Joan Manuel Marquès and Josep Jorba.

This paper presents the design of a system which allows the deployment of services in a small-sized group of computers distributed through the Internet. These groups are formed by users with a common interest, who voluntarily yield their own resources for the collaborative activities of the group. Having enough resources contributed by the members of the group, our system guarantees service availability and the fact that the deployment and execution of the services is carried out using only the resources of the group. This management is done in a completely decentralized manner, and the system is self-organized in the presence of component connections, disconnections and failures. We demonstrate its validity through the implementation of a prototype and the execution of some tests, which allow us to guarantee that the system is self-organized and always reaches a consistent state.

Introduction

When a group of people formed rather spontaneously (informal school associations, people with similar interests, campaigns for social and political activists) uses internet to carry out collaborative activities, usually they won't have supporting entities that automatically and transparently guarantee the necessary resources. These group members must thus collaborate using applications that only partially meet their needs (such as email), by having a few members manage resources for the whole group, or by paying for third-party resources or accepting advertising.

In this kind of environments, it would be desirable that the members of the group could collaborate sharing their resources. Each user should be able to provide resources so that they can be used for the benefit of the group. In this type of group the members share some common motivations and want their collaborative activities to be achieved, so they'll be willing to provide their own resources. On the other hand, as they're individual users with limited resources, variations of the capacity and availability of these resources over time can be expected, either because users disconnect, either because of components' failures, etc. Therefore, a system for managing the use of own resources in a collective way is required.

LaCOLLA [1, 2] is a middleware that allows the members of a small-sized group this kind of collaboration. Following the peer-to-peer paradigm, the group works in a decentralized manner, coordinating the resources and the applications contributed to the group by its members to allow collaboration. It basically offers general purpose functionalities that user applications can use to carry on their collaborative activities. Specifically, the middleware offers presence information, location transparency, object storage and communication between members of a group by disseminating events, which inform all the members of an action that occurred in the group, and messages, sent to a subset of participants.

When someone wants to offer a service (i.e. any application that receives a query and returns an answer) using the resources provided by the participants, whether it's an internal service for the use of the own system or one that a user wants to offer to the rest of the members of the group, its implementation should deal with all the difficulties derived from decentralized systems. A possible way to avoid this would be to extend the system with the ability to deploy a service within a group. The middleware would guarantee that the service is always active and reachable, using only resources provided by group members. This way, the service could be designed and implemented without having to manage its own decentralization, and would work in our decentralized environment. This would allow users to deploy services they want to offer but lack the resources to do it (for example, a web server), as well as provide an internal service that can be used by applications.

This paper extends the LaCOLLA middleware available at http://lacolla.uoc.edu/lacolla, and presents a system that allows the deployment of stateless services in a group of computers scattered across the Internet, in a decentralized manner, so that these services are always available as long as the resources provided to the group permit it.

Related work

There are many distributed computing systems, which allow the users to submit a task to be executed in a cluster of machines and get the result. Examples of these systems are Condor [3], Torque [12] and Sun Grid Engine [9], and also decentralized approaches like JNGI [4], built upon JXTA[5].

There are also systems which manage the deployment of web services in a distributed manner. Snap [13] creates replicas of a service on

demand, stopping these replicas when demand decreases. However, it assumes all nodes are equal and able to execute any service. We don't make this assumption, allowing nodes to participate without offering services, and services that can only be executed in very specific nodes. Another system called Chameleon [14] deploys services in a cluster while trying to maximize its "utility" (calculated from a value assigned to each service and its performance). It also assumes that any nodes can execute any service, and only one at a time, simplifying the estimation of the "utility function".

Finally, we must mention that our system makes use, when possible, of optimistic replication techniques [6], which are also used in systems like Bayou [7]. These techniques are used to share data in an efficient manner in wide area or mobile environments. Specifically, LaCOLLA uses Golding's time-stamped anti-entropy (TSAE) protocol [8].

Requirements

When defining the requirements for the design of service deployment, we must consider both the general requirements previously defined for collaboration and the concrete objectives of service deployment. Therefore, the requirements for LaCOLLA [2] also apply to this proposal. Specifically, we must consider that the system is addressed to small groups, which may be typically composed of 10 or 20 members. The main requirements considered here are the following:

- *Group self-sufficiency*: A group should not depend on external resources. Both the execution of services and the deployment management should be performed using only the resources contributed to the group by its members.
- *Decentralization and self-organization*: In case of connections, disconnections and failures, the system should keep functioning (it shouldn't have a single point of failure) and should reorganize without requiring any external intervention, getting to a consistent state as soon as the available resources and group stability allow it.
- *Individual autonomy*: The group's members should be free to decide which actions to carry out, what resources and services to provide, and when to connect or disconnect.

The requirements list is extended with the following:

- *Service availability*: A service activated by a user should always be available as long as there are enough resources to execute it in the group. This implies that there must always be an active instance of the service, executing in any of the nodes belonging gto the group, and that the users can access it. In order to achieve this, given that a service has been activated, we must consider the following aspects:
  - *Replication:* A service might be replicated to improve availability. The desired number of replicas for a service will be included in its specification. The system will try to keep that exact number of replicas, as long as there are enough available resources. Due to the optimistic approach taken, there is no absolute guarantee, and in a given moment it is possible to find a number of replicas that differs from the one specified.
  - *Location transparency*: Aplications don't have to worry about each service's locations. The system resolves them transparently, and applications access services using a location-independent identifier.

Architecture

The architecture of LaCOLLA [2] consists of five kinds of component. Each one has a different role in the general functionalities of the system, and also assumes a responsibility in service deployment. Users can instantiate the components they want to, and this determines which resources they provide to the group. This decision will be based on their degree of involvement in the group as well as their computers' capacity and availability. The different kinds of components are:

- *User Agent* (UA): Allows the interaction between applications and the system, and represents the users in the group. It allows users to create, activate, stop and access services.
- *Repository Agent* (RA): Is in charge of storing the necessary files for service execution (executables, configuration files, etc.).
- *Group Administration and Presence Agent* (GAPA): Controls the access of members to the group, and authenticates users. It acts as entry point to the system.
- *Task Dispatcher Agent* (TDA): Stores information about the services of the group. It is in charge of keeping them active and assign them to executing nodes.
- *Executor Agent* (EA): Is in charge of the execution of tasks and services.

We decided to separate the execution environments (where the services will run) from the middleware (Fig. 1). Hence, the environment is a component external to the structure of the

middleware, that connects to the group to provide some resources. The EA is in charge of communicating LaCOLLA and the execution environment, and provides an interface between them. This way, several environments with different characteristics can be connected to LaCOLLA without requiring any modification to the system. This allows services written in different languages and for different operating systems to be deployed in our system.

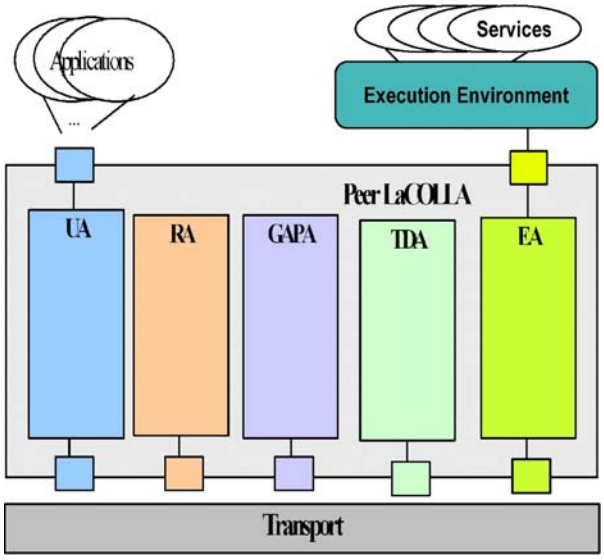

Figure 1

The API that LaCOLLA user agents offer applications includes functions for service creation, modification, deletion, activation and deactivation, as well as querying information about the existing services. In a similar way, EAs offer execution environments an API with functions for logging in and out, storing and retrieving objects from LaCOLLA and notifying exceptions. To achieve bidirectional communication, execution environments must offer an interface which allows assigning services to the environment, as well as stopping them or querying about their state and the state of the environment.

Deletion and modification of services and exception passing have not been implemented because they aren't part of the minimum subset essential to prove the viability of the proposal. Once demonstrated the feasibility of our design, it will be improved with the implementation of these functionalities.

Mechanisms

We can distinguish two kinds of mechanisms. First, the ones related to the persistent information about deployed services. These must guarantee that information about the created services is stored in a persistent and replicated way, so that it is always available and will not be lost. The second kind of mechanisms includes those related to the management of running services. The information they manage is not persistently stored, and it potentially changes quickly (e.g. location, number of replicas, etc), as they are in charge of keeping the corresponding number of active replicas of a service in the adequate nodes, ensuring its accessibility in such a dynamic environment as long as there are enough available resources.

Persistent information

The first type of mechanisms includes service creation and the propagation of this information inside the group. The components in charge of storing it persistently are the TDAs. UAs must have it cached. This requires the presence of at least one TDA connected to the group to create, activate and access services. These mechanisms use optimistic replication techniques. Propagation is accomplished in two ways:

- *Multicast*: Whenever a new information is produced (i.e., a service is created), a message is sent, using an application-layer multicast, to all the interested components (i.e., UAs and TDAs) connected, which store the information. This multicast has a moderated cost due to the small size of the groups, and allows the connected components to receive the information as soon as possible.
- *Epidemic dissemination*: Components carry out periodical anti-entropy sessions among them in order to exchange the information they know. This way, if a component misses a multicast message because of network failure, disconnection or any other reason, it will eventually receive the information through these sessions. TDAs carry out bidirectional sessions with a given number of randomly chosen TDAs. UAs, on the other hand, periodically contact a random TDA in order to update their information.

Dynamic information

The second type of mechanisms includes service activation and deactivation and, whenever possible, its guarantee of availability. Specifically, this requires at least one TDA (to manage the deployment) and one EA with a connected environment which matches the service's specified requirements.

To carry out this management, a service is assigned, in the moment of activation, to a TDA which will act as master. It periodically checks that the current number of replicas is the appropiate, detects stopped service replicas, and stops or activates new replicas when needed. In addition, there is a set of secondary masters assigned to each service, chosen by the primary master, which maintain the information about

the service management kept by the primary, and can take its place in case it fails. LaCOLLA provides a presence mechanism, so that TDAs don't need to obtain this information and will automatically detect other TDA's failure. It must be considered, though, that because of the optimistic approach taken in the mechanisms' design, this detection is not immediate.

The master periodically multicasts the information about who the primary and the secondaries are to every connected TDA and UA. The UAs also receive the list of locations of the service.

The master of the service is in charge of choosing the nodes where the service will be executed. In order to do this, it checks the specification of each of the execution environments connected to the group, comparing them with the requirements of the service. With the ones that match the required specification, the master creates a list of candidate execution environments. Among these, it chooses the number corresponding to the desired number of replicas for the service. The TDA sends a message to every chosen environment, containing the specification of the service and telling them to execute it. On reception, each environment will get the necessary files from the storage provided by LaCOLLA (i.e., the RAs) and start execution. Periodically, the TDA will send messages to the environments, to remind them that they have to execute the service. If an environment doesn't receive these messages for a certain time interval, it will consider that he doesn't have to execute the service any longer, and will stop it. This guarantees that, in case of master failure or disconnection, if the information about the locations of the service is lost, no unknown and hence unavailable replicas will keep executing. To ensure the TDAs know the environments connected to the group, they carry out periodical synchronization sessions with the EAs. Also, EAs use multicast messages to inform TDAs of execution environment's connections and disconnections.

Two different mechanisms have been designed to elect a master for the service. One is inspired in Distributed Hash Tables (DHT), while the other is based on CSMA/CD, the technique used in Ethernet.

The pseudo-DHT method is based on the main idea of DHTs, assignation of the responsibility on a particular entity to a certain node based on a proximity function. We obtain keys by hashing the service identifier and each TDA identifier, using the function SHA-1 [10]. Resembling Chord [11], the TDA keys are organized in a ring, so that for each service, the closer TDA is the one with the same key or the first successor. Unlike DHTs, we don't need to have a service always assigned to the closest node, due to the master disseminating information periodically. Because of that, as long as the master for a service is active, the connection of new members will not change this assignation. This way we save ourselves to make many responsibility shifts if the level of churn in the group is high. When the master disconnects or fails, each of the secondary masters of the service will check if they're the next successor. If this is the case, that TDA will proclaim itself as primary master. In case there are no active secondary masters, the primary will be elected among the rest of the TDAs. If conflicts arise from the decisions taken locally by the TDAs due to the eventual consistency of the presence mechanism (in a given moment, a component may not know all connected members, or may think a disconnected member is still connected), they will be eventually detected (using the presence mechanism and the multicast messages sent by the master) and solved using the proximity function.

The pseudo-CSMA/CD method takes advantage of randomness to distribute the responsibilities on services among the TDAs. Firstly, when a service is activated, the UA who receives the request randomly chooses a TDA, which is assigned as the master. Likewise, this one chooses the secondaries randomly. The problem comes when a disconnected master must be substituted. This is where the CSMA/CD mechanism is taken as model. We used its negotiation mechanism, based on random intervals, to design a negotiation method between TDAs to decide who is responsible of a service. When a TDA detects that the master of a service is no longer available, if it was one of the secondary masters of the service, or wasn't but doesn't know of any active secondary, it can decide to substitute it. In case this decision is favorable, it informs the rest of the TDAs. After this, it waits a certain interval for any equivalent message from another TDA (which, in CSMA/CD, would mean a collision). In case it is received, the TDA will wait a random interval. If, during this wait, it receives a second message of the other TDA, it will withdraw from the negotiation and yield the responsibility to the other one. Otherwise, it will send the message again, waiting for another interval to detect messages from other TDAs. In case it doesn't receive any, it will infer that he has won the negotiation and will proclaim himself as the master, multicasting a message to all the TDAs.

Finally, when a user wants to access a service, he asks its location to the UA where he is connected and uses it to directly contact the service. In case that, for any reason, the UA doesn't have this information, it will ask the master of the service or any of the secondaries, and provide it to the user.

Validation

We have a prototype of the LaCOLLA middleware implemented in Java that can work either in real time or in simulation time (measured in iterations). It offers an interface for the users to access the functionalities of the middleware, and also has the ability to simulate user's activity and system dynamism (connections, disconnections, failures) in order to conduct tests and validate its functioning.

The basic core of the service deployment system has been implemented on this prototype. Specifically, we have implemented the mechanisms for service creation, activation, deactivation and access. Both methods for master election, pseudo-DHT and pseudo-CSMA/CD, have been implemented.

This implementation has been used to carry out tests with groups of 10 components, formed by one RA, one GAPA, two UAs, four TDAs, and two EAs.

Tests have been carried out in order to verify that the system does converge, that is, it reaches a consistent state. Each experiment was divided into two phases: during the first one, we simulated user's activity (service creation, activation and deactivation) and system dynamism (failure and disconnection of components); the second phase involved only internal mechanisms. We measured the number of steps it took LaCOLLA to achieve consistent information in all the components.

Table 2 shows component disconnection and failure probability per iteration, while table 3 shows the probability of activity generation.

	Failure		Disconnection	
	Prob.	Duration	Prob.	Duration
UA and EA	0.0005	[15,120]	0.0025	[60,540]
RA, GAPA and TDA	0.000125	[12,60]	0.0005	[15,120]

Table 2

Action	Probability per iteration
Service creation	0.01
Service activation	0.007
Service deactivation	0.005

Table 3

These tests have been carried out for both the pseudo-CSMA/CD method and the pseudo-DHT method. Figure 2 shows the cumulative probability that in N iterations the system has reached consistency using both methods. As can be seen in the graphs, in both cases the system reaches a consistent state in few iterations. Specifically in an 85% of the executions the system is consistent in the same moment the first phase ends, and in only two iterations 95% of the executions have already reached consistency.

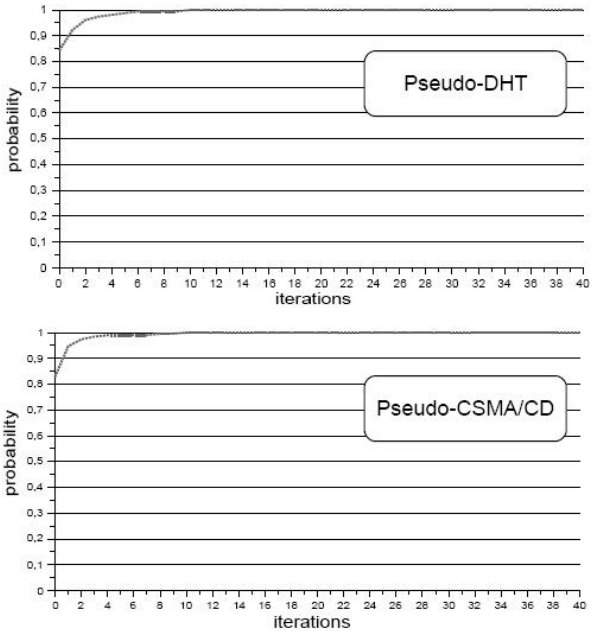

Figure 2

From the results we deduce that the design proposed makes the system reach a consistent state, proving that the decentralized management and self-organization work in presence of churn. With current tests, though, we can not observe important differences between the behaviors of the two mechanisms implemented for master election. Therefore, we can say that both are valid for the operation of the system, although it will be necessary to carry out more exhaustive tests in order to identify in which situations and environments we can obtain better performance of one particular mechanism.

Conclusions and future work

We have presented a basic core for the deployment of stateless services in a collaborative environment. However, there are many aspects that can be further developed. Future lines of work include:

- Carry out more exhaustive tests, using a greater number of components and collecting different measurements.
- Add the functionalities for service modification and elimination, while keeping the system's self-organization capability, so that it can solve by itself the possible inconsistencies caused by concurrent updates.
- Allow the deployment of stateful services.

- Allow communication between different replicas of a service, through message passing among them. This possibility would allow offering more complex services, where the existing replicas coordinate their actions.
- Build execution environment able to execute other types of applications (current implementation can only execute Java applications), adding new considerations like security, isolation from the machine where it's running, users privacy, etc. This would allow its use in environments where a trust relation between users does not exist, and hence extend the system's usability.
- Explore possible modifications of the proposed mechanisms to improve system's scalability, so that it can be applied to large groups.

References

[1] Marquès, J.M."LaCOLLA: Una Infraestructura Autònoma i Autoorganitzada per Facilitar la Col·laboració" [LaCOLLA: An Autonomous and Self-Organized Infrastructure to Facilitate Collaboration], doctoral thesis, Dept. of Computer Architecture, Tech. Univ. of Catalonia, 2003, http://people.ac.upc.es/marques/LaCOLLA-tesiJM.pdf.

[2] Marquès, J.M., Navarro, L., Vilajosana, X., Daradoumis, A. (2007). LaCOLLA: Middleware for Self-Sufficient Online Collaboration. IEEE Internet Computing. pp. 16-24. ISSN:1089-7801. (to be published on January 2007)

[3] Thain, D., Tannenbaum, T., Livny, M. Distributed Computing in Practice: The Condor Experience. Concurrency and Computation: Practice & Experience. v. 17, Issue 2-4 (February 2005) pp. 323-356 ISSN:1532-0626

[4] Verbeke, J., Nadgir, N., Ruetsch, G., and Sharapov, I. Framework for peer-to-peer distributed computing in a heterogeneous, decentralized environment. In Proceedings of the 3rd InternationalWorkshop on Grid Computing, November 2002.

[5] Gong, L. JXTA: A network programming environment. IEEE Internet Computing, v. 5, pp. 88-95, 2001

[6] Saito,Y., Shapiro, M. (2005). Optimistic replication. ACM Computing Surveys (CSUR). v.37 n.1, p.42-81, March 2005.

[7] Terry, Douglas B., et al. Managing Update Conflicts in Bayou, a Weakly Connected Replicated Storage System. Proc. 15 th Symposium on Operating Systems Principles, Dec 1995

[8] Golding, R.A. Weak-consistency group communication and membership, USCS-CRL-92-52, Concurrent Systems Laboratory, University of California, 1992.

[9] http://gridengine.sunsource.net

[10] NIST,"Secure hash standard", U.S. Department of Commerce, National Technical Information Service FIPS 180-1, Apr. 1995.

[11] Stoica, I., Morris, R. et al. Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications, IEEE/ACM Trans. Net., vol. 11, no. 1, 2003, pp. 1732.

[12] http://www.clusterresources.com/pages/products/torqueresource-manager.php

[13] Pairot, C., García, P., Mondéjar, R., "Deploying Wide-Area Applications Is a Snap," *IEEE Internet Computing*, vol. 11, no. 2, pp. 72-79, Mar/Apr, 2007.

[14] C. Adam and R. Stadler, "Implementation and Evaluation of a Middleware for Self-Organizing Decentralized Web Services," *Proc. IEEE SelfMan 2006*, IEEE CS Press, 2006.

Daniel Lázaro, Joan Manuel Marquès and Josep Jorba

Open University of Catalonia

Research Project Focus: Grid4All

Grid4All is an EC project aimed at helping to bring global computing to the broader society beyond that of academia and large enterprises by providing an opportunity to small organizations and individuals to reap the cost benefit of resource sharing without, however, the burdens of management, security, and administration.

Grid4All embraces the vision of a "democratic" Grid as a ubiquitous utility whereby domestic users, small organizations and enterprises may draw on resources on the Internet without having to individually invest and manage computing and IT resources. This project funded by the 6° Framework Programme of the European Commission involves institutional and industrial partners [1].

Project Objectives

Grid4All will achieve the following objectives for the Grid community:

- Alleviate administration and management of large scale distributed IT infrastructure - by pioneering the application of component based management architectures to self-organizing peer-to-peer overlay services.
- Provide self-management capabilities - improve scalability, resilience to failures and volatility, thus paving the way to mature solutions enabling deployment of Grids on the wide Internet.
- Widen the scope of Grid technologies by enabling on-demand creation and maintenance of dynamically evolving scalable virtual organizations, even short lived.
- Apply advanced application frameworks for collaborative data sharing applications executing in dynamic environments.
- Capitalise on Grids as revenue generating sources to implement utility models of computing..

Project Work Packages

WP1: Overlay Infrastructure and Programming Models

The objective of WP1 is to provide middleware and services provided by the other work packages with the basic infrastructure, models and tools to realize the Grid4All vision. In other words this WP provides the architectural model, framework and core services of the platform. Based on peer-to-peer overlay network technology and component model, WP1 provides framework (tools, APIs, and programming models) for the construction of self-managing peer-to-peer applications. It also provides communication, routing, messaging and lookup functionalities and infrastructure support for the construction of reliable and self-managing services and applications, including that of creation and management of Grid4All virtual organizations (mapped as an overlay service).

Furthermore WP1 will provide the models and tools to allow system and application developers to easily use the loosest and weakest synchronization and consistency models consisted with system and application goals. In particular a framework to understanding and instrument suitable replication protocols based on the Action-Constraint Framework (ACF) will be developed.

WP2: Virtual Organization and Resource Management

Grid4All proposes a framework based approach for the creation and management of virtual organizations. Within Grid4All virtual organizations (VO) are run time entities federating resources, services and members. VOs encapsulate all management functions such as resource allocation, membership management, performance management, monitoring, communication and security, simplifying the construction of dynamic and distributed virtual communities.

The objectives of this WP are to provide an autonomic management framework for dynamic virtual organizations spanning multiple management domains and to develop supporting tools for the construction of Grid4All self-management domains.

WP2 will provide middleware and VO management services. It will use the framework and core services provided by WP1 to build higher level Grid management functions as overlay services: Creation, deletion and management of dynamic virtual organizations. Control of resources, services and users forming the virtual organization. Service/resource discovery based on semantic descriptions of Grid4All entities

Market based resource management for provisioning of resources and services to virtual organisations. An implementation of a scheduling service – deployed within a virtual organization.

WP3: Data Storage

The Grid4All usage scenarios, centered on collaboration and utility computing for communities, small enterprises and individuals call for simple, secure and resilient access to shared data. Distributed access to shared data may occur on a large scale. Applications such as multimedia, collaboration tools and e-learning systems have specific and non-traditional storage needs. Furthermore, the storage mechanisms need to be adapted to the VO concept and to run above the large-scale, dynamic, heterogeneous environment of Grid4All.

Within the Grid4All structure, WP3 provides data services constructed using self-managing software framework provided by WP1. Work performed within WP3 plan to implement a distributed file service for a large scale heterogeneous Grid and to export a storage infrastructure built upon secure site-wide storage pools. It is also planned to federate views exported from different organisations into a unified file system for a Virtual Organization across management domains. Finally a semantic storage architecture will be explored for a collaborative work and at large scale.

WP4: Applications

WP4 provides both the applications and directs the use of scenarios. This WP provides not only the implementation of proof-of-concept applications, but also the framework and API permitting the development of new applications.

This WP will have to determine scenarios and derive requirements for infrastructure and applications specific for the Grid4All intended uses. Partners involved will have to understand the complexity imposed by this environment on applications and to define algorithms to overcome situations and limitations. WP4 also plans to define and develop an API and selective transparency mechanisms to facilitate application development. The last action within this WP will be to produce application level traces and benchmarks to evaluate and validate the Grid4All infrastructure.

WP5: Integration and Evaluation

WP5 is in charge of:

Integrating the different middleware building blocks developed within the work packages WP1-WP4 in an iterative fashion in order to build the Grid4All middleware infrastructure. This includes the production of guidelines, packaging and configuration tools.

Defining an evaluation plan for the platform that identifies relevant performance metrics derived from the requirement analysis phase performed by WP1-WP4. This plan will also detail how the Grid4All system will be hosted on the test-bed(s) selected for the evaluation.

Deploying the Grid4All middleware infrastructure along with the applications developed in WP4 on the chosen test-beds.

Evaluating the Grid4All platform on the basis of evaluation plan and providing feedback to technical WP (WP1-WP4) in order to improve the platform both from a functional and performance point of view.

WP6: Dissemination and Exploitation

This WP sets the basis for the execution of a good dissemination campaign of the project and for the exploitation of the obtained results. A preliminary awareness about the project will be created and a set of dissemination actions will be developed. These actions will be developed by the whole consortium in order to drive the project to a Europe-wide dissemination. An Exploitation Plan will be created defining the actions that will be jointly taken by the consortium partners following the project to further the exploitation of the results. Communication and public relations addressing the different category of target audience (end users, service providers, VARs, application providers) will be a part of this WP. This also includes identification and participation in public events where Grid4All can be demonstrated.

Actions developed within this WP are: Dissemination and technology transfer plan definition, establishment of dissemination standard, quality assurance procedure, integrated project web site and e-newsletter, organisation of events and relation with other media, Grid4All contacts database, establishment of external collaborations and communication and public relations.

References

[1] Project Web Site: http://www.grid4all.eu.

J.M. Marquès, X. Vilajosana, T. Daradoumis
DPCS Research group

Open University of Catalonia

Technology Watch

Between Discovery and Matching of learning Grid Services: a way to take advantage of Semantic Capabilities.

Grid Learning Services Description

WSDL (Web Service Description Language) describes the functional information of services such as input parameters, output parameters, service providers and service locations. However, it is limited in supporting the discovery, execution, composition and interoperation of Web services. WSDL cannot provide semantic information of Web services that enable the semantic description of services capabilities.

Currently Globus Toolkit is a common way to implement Grid Services. Globus Metacomputing Directory Service (MDS) implements a standard Web Services interface to a variety of local monitoring tools. Thus, within Globus Toolkit, MDS allows one to register Grid services. Besides it, UDDI has been also used in the web community for business service discovery. Both of them only support keyword based search and are limited in semantic description.

OWL-S is a representative semantic Web service language that arises from the standardization of DAML-S, by integrating OWL-based ontology technology with existing Web service description.

WSMO (Web Service Modeling Framework) provides ontological specifications for the core elements of Semantic Web Services. In fact, Semantic Web Services aim at an integrated technology for the next generation of the Web by combining Semantic Web technologies and Web Services, thereby turning the Internet from an information repository for human consumption into a world-wide system for distributed web computing.

BPEL4WS (Business Process Execution Language for Web Services) provides a language for specifying business processes and business interaction protocols. It can create a composite process by integrating different operations such as Web service call, data manipulation, error report, and process termination.

Nevertheless, these technologies are still immature and incomplete. Moreover, they compete each other; in fact, they still do not provide viable and integrated solutions to the web Services discovery problem.

[1] Foster. Globus Toolkit Version 4: Software for Service-Oriented Systems. IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779, pp 2-13, 2005.

[2] W3C, OWL-S, http://www.w3.org/Submision/-OWL-S/.

[3] W3C, Web Service Modelling Ontology (WSMO), http://www.w3.org/Submission/WSMO/.

[4] F. Curbera et al., Business process execution language for web services (BPEL4WS) 1.0, July 2002, http://www-106.ibm.com/developerworks-/webservices/library/ws-bpel.

Discovery of Grid Learning Services

Discovery is the process of finding Web services with a given capability. In general, discovery requires that Web services advertise their capabilities with a registry, and that requesting services query the registry for Web services with particular capabilities. The role of the registry is both to store the advertisements of capabilities and to perform a match between the request and the advertisements.

In general, a semantic discovery process relies on semantic annotations, containing high-level abstract descriptions of service requirements and behavior. Metadata is an essential element in semantic discovery with the capability to expand service descriptions with additional information. The achievement of dynamic composition and automation of services involves discovering new services at run time by software components without human interaction. SOAP provides a description of message transport mechanisms, whereas WSDL describes the interface used by each learning service. However, neither SOAP nor WSDL are of any help for the automatic location of learning services on the basis of their capabilities. Paolucci comments that in order to enable the automation of this process we need a meaningful description of the service and its parameters that can be processed automatically by tools. This implies the possibility to process the context of description by discovery engines.

In this sense, there are some works that aim to improve the semantic services capability of matching. On the one hand, in Paolucci focuses primarily on comparing inputs and outputs of a service as semantic concepts represented in OWL to improve UDDI. This work proposes a way of ranking semantic matching results. This ranking can be used in conjunction with other

user-defined constraints to inform of an exact, or potentially useful web-service capability match. On the other hand, there are important lines of research that propose extensions to Web service description WSDL in two ways, annotated WSDL and WSDL-S files. These approaches try to adhere to the current standards while trying to maximize semantic representations required for automation.

References

[5] Massimo Paolucci, Takahiro Kawamura, Rerry R. Payne, and Katia Sycara. Semantic matching of web services capabilities, The Semantic Web - ISWC 2002: First International Semantic Web Conference, Sardinia, Italy, June 9-12, 2002.

[6] Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, Katia Sycara, Importing the Semantic Web in UDDI, Robotics Institue, Carnegie-Mellon University, USA, 2002.

Grid Learning Services Matching and Composition

There are three principal motivations for Learning Grid Services Composition: build a more powerful service using basic existing services, fulfil service requester's requirement better, and enhance resource reuse while reducing the cost and time of a new service development.

In general, a framework used for Web service composition (Fig. 1) describes two kinds of participants, service provider and service requester. It contains the following components: a translator, a process generator, an evaluator, an execution engine and a service repository.

The service providers propose Web services for use. The service requesters consume information or services offered by the service providers. The translator translates between the external languages used by the participants and the internal languages used by the process generator. For each request, the process generator tries to generate a plan that composes the available services in the service repository to fulfill the request. If more than one plan is found, the evaluator evaluates all plans and proposes the best one for execution. The execution engine executes the plan and returns the result to the service provide.

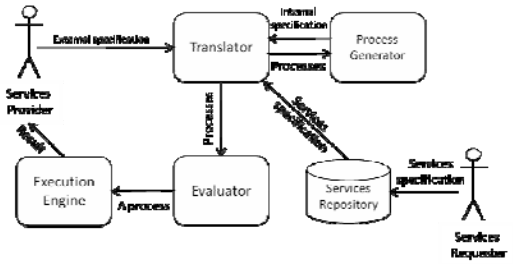

Fig 1. The framework of the service composition system

IMS Global Learning Consortium proposes an abstract framework representing a set of services used to construct an e-learning system in its broadest sense. Fig 2. shows the dependencies between the different "layers" of the framework.

The Learning Application composition process consists of identifying sub-tasks of the learning process, locating suitable Learning application Services to construct each process, locating suitable Common Services to construct each learning service, formatting the Learning and Common services into a service flow and executing the service flow to achieve a task which is the goal of the learning process.

The core stage is the composition of learning web services and their adaptation to the needs of a learner or group of learners. Such a composition is carried out by retrieving previously registered objects. Once composed and packaged as learning objects, these composite processes can be executed and then instantiated and adapted to the learner's particular needs.

These adaptations can be realized, either by predefined rules implemented into the process description and driven by the learner behavior, or in a supervised manner. In the later case, the instructional designer can return to the composition tools to adapt the process.

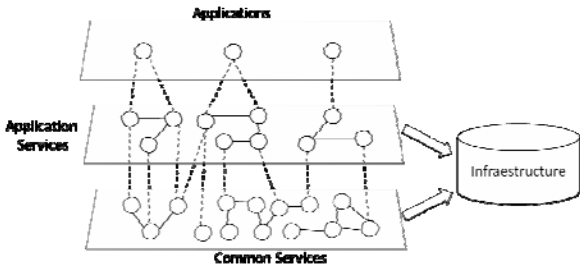

Fig 2. IMS Abstract Framework

Dealing with the specific problem of constructing a suitable workflow for a learning scenario, in [9] the authors propose a framework to facilitate automated composition of scientific workflows in Semantic Grids made up of a Manager Service and other supporting services, including an abstract and a concrete workflow generator. They described these components in detail and outlined their interactions. Finally, they described their implementation and its use within the physics domain. The important features of this approach are: an adaptive workflow generation algorithm and the distinction between different levels of abstraction of the workflow in order to allow reuse and sharing.

Furthermore, there is a detailed development of learning services matching procedures for locating the most suitable Learning Services, combining and integrating a number of matching algorithms, and adopting two principal approaches: the structural matching approach and the linguistic or syntactic approach. This work focuses on the issue of searching a Web Service with required functionalities and addressing a specific application domain, by means of an ontology-based semantic description.

References

[7] Jinghai Rao and Xiaomeng Su, A Survey of Automated Web Service Composition Methods, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, ISSN 0302-9743, Volume 3387, 2005.

[8] Jorge Torre, Juan Manuel Dodero, Carmen L. Padrón, A Framework Based on Web Services Composition for the Adaptability of Complex and Dynamic Learning Processes, Learning Technology newsletter, Vol. 6, Issue 1, January 2004.

[9] Shalil Majithia, David W.Walker, W.A.Gray, Automated Composition of Semantic Grid Services, International Conference on Autonomic Computing (ICAC'04), May, 2004

[10] Ching-Jung Liao, Fang-Chuan Ou Yang, A Workflow Framework for Pervasive Learning Objects Composition by Employing Grid Services Flow Language. Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT'04), pp: 840 – 841, ISBN:0-7695-2181-9, 2004.

[11] Xiaofeng Du, William Song, Malcolm Munro, Service Composition in the Context of Grid, AHM2006, 2006

Gustavo Gutiérrez

Universidad Michoacana de
San Nicolás de Hidalgo

News

By Josep Jorba

IBERGRID: 1st Iberian Grid Infraestructure conference

14 May 2007

On November 2006, the Presidents of the governments of Portugal and Spain agreed to adopt a series of policies and actions toward the creation of a Iberian Research Area. To this end, it was decided to connect the research networks of both countries and to deploy an Iberian Grid Infrastructure in order to better serve the computing needs of the Iberian Research community. Said infrastructure would later adopt the name of IBERGRID.

As an additional measure to deepen collaborative efforts between both countries, an annual conference is to be held. It was decided that the first of these annual conferences would be held in Santiago de Compostela, Spain, in May, 2007.

The conference aims to foster and promote R&D activities in the Iberian States and their links to Latin America by bringing together academics and industry agents that are cooperating in computer and computational sciences applied to grid infrastructure and technologies.

The presentations, also the papers in proceedings volume, include key Grid protagonists, including an Opening Talk by Ian Foster about Scaling eScience Impact. Others talks, was focused in different key projects, like EGEE Project, EELA (a project shared between Europe and Latin America), CyTEDGrid (also in iberoamerican area). Participants were able to see also different technical sessions about operational grid demos (in EGEE, and DEISA), and details on different software support for Grid projects, like EEGE accounting mechanisms, SGE schedulers, Virtualization for grid deployment, gLITE scheduling strategies.

Also some national grid projects are presented, as German D-Grid, Greek Hellasgrid, and Scandinavian countries NorduGrid.

For further information:

http://www.ibergrid.eu/

CyTEDGrid: Grid Technology for Boosting Regional Development

2006-2008

CyTEDGrid is an Ibero American project funded by CyTED (Ciencia y Tecnología para el Desarrollo - Science and Technology for Development) for three years included on its Area 5: Information and Communication Technologies.

The general purpose of the project is to build a human and technical framework around Grid technology to foster collaboration among a wide number of academic Ibero American groups interested in parallel computation.

The main specific goal is to deploy a hardware and software Grid infrastructure in order to make it available to non-computation specialist researches in the Ibero American community, and to generate technical capabilities and knowledge in the Grid computation area among the participants groups, paying special attention to the improvement of human resources.

For further information:

http://www.cytedgrid.org/english/introduction.html

XtreemeOS Project

2006-2010

The overall objective of the XtreemOS project is the design, implementation, evaluation and distribution of an open source Grid operating system (named XtreemOS) with native support for virtual organizations (VO) and capable of running on a wide range of underlying platforms, from clusters to mobiles.

The approach we propose in this project is to investigate the construction of a new Grid OS, XtreemOS, based on the existing general purpose OS Linux. A set of system services, extending those found in the traditional Linux, will provide users with all the Grid capabilities associated with current Grid middleware, but fully integrated into the OS. The underlying Linux will be extended as needed to support VOs spanning across many machines and to provide appropriate interfaces to the Grid OS services.

Installed on each participating machine, the XtreemOS system will provide for the Grid what an operating system offers for a single computer: abstraction from the hardware, and secure resource sharing between different users. It would thus considerably ease the work of users belonging to VOs by giving them (as far as possible) the illusion of using a traditional computer, and releasing them from dealing with the complex resource management issues of a Grid environment. By integrating Grid capabilities into the kernel, XtreemOS will also provide a more robust, secure and easier to manage infrastructure for system administrators.

The XtreemOS consortium composition is a balance between academic and industrial partners interested in designing and implementing the XtreemOS components (Linux extensions to support VOs and Grid OS services), packaging and distributing the XtreemOS system on different hardware platforms, promoting and providing user support for the XtreemOS system, and experimenting with Grid applications using the XtreemOS system. Different end-users are involved in XtreemOS project, providing various test cases in scientific and business computing domains.

When	What	Where
September 6-8, 2007	**Workshop in GridComputing in 2007 IEEE International Conference on Intelligent Computer Communication and processing** The workshop aims to bring together computer scientists, and experts who are working in the field of grid computing. The workshop consists of invited papers from internationally recognized experts presented in a half day session. For more information: http://transilvania.coned.utcluj.ro/~asuciu/wgc	**Cluj-Napoca, Romania**
September 10-14, 2007	**International GridKa School** GridKa School, now in its fifth year, is a computing school covering topics related to Scientific Grid Computing. It is targeted at everyone involved in grid computing, from beginners to administrators: * scientists from different disciplines (especially advanced undergraduate, graduate students and postdocs), * members of the various national and international grid projects. Topics for the school are likely to be chosen from the realm of the EGEE and D-Grid initiatives and beyond. For more information: http://iwrwww1.fzk.de/gks07/	**Forschungszentrum Karlsruhe, Germany**
October 1-5, 2007	**EGEE'07 (Enabling Grids for E-science)** EGEE'07 is the latest in a series of prestigious conferences which regularly attract over 600 key players from the international Grid user communities, decision makers, resource providers, developers, governments and businesses. EGEE'07 provides a platform to discuss connections with different community users and related projects, and how to drive forward world-class Grid technologies. The program will see keynote speakers and EGEE presentations together with parallel sessions on focused tracks: * The innovative business track introduced at the last conference in Geneva, Switzerland, will be renewed this year, giving the business community an opportunity to understand how to benefit from Grids and adopt Grid technologies. Companies are invited to take up the possibility to have a stand at the conference exhibition; * The application track will see technical discussions on what techniques are best suited for applications and what issues applications still face on the EGEE infrastructure following the input received at the 2nd EGEE User Forum. A demo and poster area complements the application track and a special drink in the demo area will enhance the networking possibilities; * Various collaborating projects will interact with EGEE and within their community in the collaborating projects track. A sustainability workshop will be one of the highlights of this track. Collaborating projects are also invited to take part of the conference exhibition; * Finally, the EGEE activities will coordinate in the cross-activity track which is complemented by an EGEE federations track that provides coordination facilities for the EGEE federations. For more information: http://www.eu-egee.org/egee07/programme	**Budapest, Hungary**

When	What	Where
October 17-19, 2007	**GridNets 2007 (International Conference on Networks for Grid Applications)** The GridNets conference series is an annual international meeting which provides a focused and highly interactive forum where researchers and technologists have the opportunity to present and discuss leading research, developments, and future directions in the Grid networking area. The objective of this event is to serve as both the premier conference presenting best Grid Networking research and a forum where new concepts can be introduced and explored. Grid developers and practitioners are increasingly realizing the importance of an efficient network support. Entire classes of applications would greatly benefit by a network-aware Grid middleware, able to effectively manage the network resource in terms of scheduling, access and use. Conversely, the peculiar requirements of Grid applications provide stimulating drivers for new challenging research towards the development of Grid-aware networks. Cooperation between Grid middleware and network infrastructure driven by a common control plane is a key factor to effectively empower the global Grid platform for the execution of network-intensive applications, requiring massive data transfers, very fast and low-latency connections, and stable and guaranteed transmission rates. Big e-science projects, as well as industrial and engineering applications for data analysis, image processing, multimedia, or visualization just to name a few are awaiting an efficient Grid network support. They would be boosted by a global Grid platform enabling end-to-end dynamic bandwidth allocation, broadband and low-latency access, inter-domain access control, and other network performance monitoring capabilities. For more information: http://gridnets.org/2007/	**Lyon, France**
November 29-30, 2007	**International Conference on Grid computing, high-performance and Distributed Applications (GADA'07)** the GADA workshop arose in 2004 as a forum for researchers in grid computing whose aim was to extend their background in this area, and more specifically, for those who used grid environments in managing and analyzing data. Both GADA'04 and GADA'05 were constituted as successful events, due to the large number of high-quality papers received, as well as the brainstorming of experiences and ideas interchanged in the associated forums. Because of this demonstrated success, GADA was upgraded as a Conference within On The Move Federated Conferences and Workshops (OTM'06). GADA'06 covered a broader set of disciplines, although grid computing kept a key role in the set of main topics of the conference. The main goal of GADA'07 is to provide a framework in which a community of researchers, developers and users can exchange ideas and works related to grid, high-performance and distributed applications and systems. The second goal of GADA'07 is to create interaction between grid computing researchers and the other OTM attendees. GADA'07 intends to draw a highly diverse body of researchers and practitioners by being part of the "On the Move to Meaningful Internet Systems and Ubiquitous Computing 2007" federated conferences event that includes five co-located conferences: GADA'07, CoopIS'07 (International Conference on Cooperative Information Systems), DOA'07 (International Symposium on Distributed Objects and Applications), ODBASE'07 (International Conference on Ontologies, Databases, and applications of Semantics) and IS'07 (Information Security Symposium). For more information: http://www.cs.rmit.edu.au/fedconf/index.html?page=gada2007cfp	**Algarve, Portugal**

When	What	Where
December 10-13, 2007	**3rd IEEE International Conference on e-Science and Grid Computing** The next generation of scientific research and experiments will be carried out by communities of researchers from organizations that span national boundaries. These activities will involve geographically distributed and heterogeneous resources such as computational systems, scientific instruments, databases, sensors, software components, networks, and people. Such large-scale and enhanced scientific endeavors, popularly termed as e-Science, are carried out via collaborations on a global scale. Grid computing has emerged as one of the key computing paradigms that enable the creation and management of Internet-based utility computing infrastructure, called Cyber infrastructure, for realization of e-Science and e-Business at the global level. To harness the potential of e-Science and Grid computing paradigms, several national and international projects around the world have been initiated to carry out research and innovation activities that transform the goal of e-Science and Grid computing into a reality. The e-Science 2007 conference, sponsored by the IEEE Computer Society's Technical Committee for Scalable Computing (TCSC), is designed to bring together leading international and interdisciplinary research communities, developers, and users of e-Science applications and enabling IT technologies. The conference serves as a forum to present the results of the latest research and product/tool developments, and highlight related activities from around the world. Some topics concerning e-Science and Grid Computing are of interest, but not limited to: * Enabling Technologies: Internet and Web Services * Collaborative Science Models and Techniques * Service-Oriented Grid Architectures * Problem Solving Environments * Application Development Environments * Autonomic and Self-Organizing Grid Networks * Security Challenges * Software and Social Engineering For more information: http://www.garudaindia.in/e-science_2007.asp	**Bangalore, India**