Grid and Portable and Multimedia Resources

Asignatura: Aplicaciones Distribuídas Avanzadas

Omar Aurelio Walid Llorente <omar@dit.upm.es>

June, 2004

Grid computing and infraestructure is being widely deployed in research, commercial and educational environments with the intention of taking advantage of the big amount of computers interconnected in campuses and enterprises. These computers, besides of being connected through LANs and MANs, have a huge amount of resources (CPU time, CPU power, bandwidth, graphical processors, etc) that may be very useful for the implementation of many kinds of distributed services. At the same time, there are a big number of initiatives to give portable computers, cell phones and PDAs the opportunity of being not only portable, but wireless, wearable and multimedia (MM). With this article the author will try to give some clues about the way this two environments (Portable/MM and Grid computing) may be combined to form a powerful and practical solution.

1 Introduction to General Grid Concepts

Grid is generally defined as a "system framework into which hardware or software components can be plugged, and which permits easy configuration and the creation of new functionality from existing components". This kind of systems are usually related to applications where big amounts of CPU cycles, communication bandwidth or data, where customized computing services universally available or in environments where sharing and cooperation of dynamically distributed resources are needed.

Can be said that Grid is a concept or mechanism where the focus is on integrating a set of distributed resources that can applied at different computer system levels: computation, data, software, agents, users, etc. There are many fields where Grid-like ideas can be used. For example, **computational grids** intend to serve as platform for supercomputing applications, **computing fabrics** are intended to provide ubiquitous computing capabilities, and **agent-grids** try to form a system where the computational grid issues, the associated metadata, the distributed computation and load balancing, the mobile code, the security, etc., as well as the "agent-level" versions of those issues are involved.

In computational grids, can be mentioned as one of the biggest efforts, the case of the European DataGrid Project, which aims at "developing, implementing and exploiting a large-scale

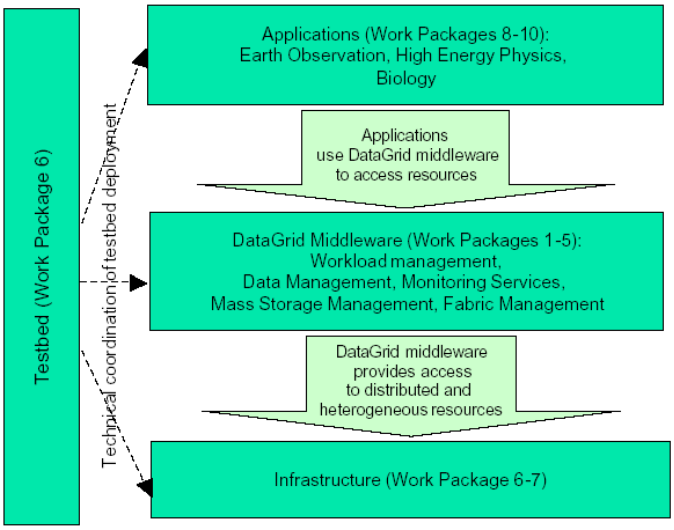

Figure 1: Organization of the technical layers in the European DataGrid Project

data and CPU-oriented computational grid, allowing the geographically distributed processing of huge amounts of data coming from three different scientific disciplines: High Energy Physics, Biology and Earth Observation". The major concerns of this sharing technology are complicated by the distributed nature of the resources to be used, the dispersed communities of scientists, the size of the databases and the limited network bandwidth.

For allowing the integrated computing and data access, is needed a big homogenization effort, provided usually by specific middleware that functions as an intermediate application layer used to access in an uniform manner all the distributed and different resources, as if they were a unique facility. This way, Grid computing infrastructures have been able to demonstrate that are capable of managing testbeds with many Peta-bytes of distributed data, tens of thousands of resources (processors, disks, ...), and thousands of simultaneous users, allowing the scientific exploration of high-complex and big-data producer tasks to take a step forward.

Among the capabilities provided by the middleware layer, there are such important ones as: workload and data management, monitoring services, mass storage management and fabric management. The middleware layer is, of course, responsible of attending the requests of the applications while using the dynamically available Grid infrastructure for obtaining the real resources needed by them. In 1 can be seen the real layer deployment for the DataGrid Project.

The basic concept of a computational grid is used to indicate an analogy with the electrical power grid, because as a power grid is used to link different sources of electrical power, and provides for widespread access to and distribution of that power, a computational grid is "a software and hardware infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities". In the growing environment of computational grids, a number of initiatives can be mentioned as the more suitable and deployed:

- Distributed supercomputing: mainly used in distributed interactive simulations (like mili-

tary ones), and simulation of physical processes (e.g., stellar dynamics and climate modeling) for which large amounts of resources are needed.

- High-throughput computing: used to solve cryptographic or complex design problems that can be divisible into a large number of parallel, loosely-coupled or independent tasks.
- On-demand computing: applications usually driven by cost-performance concerns rather than absolute performance, and having to do with large population of users and resources in fault-tolerance, payment and security environments.
- Data-intensive computing: for synthesizing distributed data repositories by means of processes often computationally and communication intensive. DataGrid Project, previously introduced may be a good example.
- Collaborative computing: activities related with human-to-human interactions (mainly using virtual worlds, virtual communication paths or heavy CPU-consumer multimedia streaming services).

Computing fabrics, however, are described as sets of nodes, which are packages of processors, memory, and peripherals, linked together by an interconnection facility. Depending on the tightness of the coupling between nodes, some of them can be called cells, which can even divide or merge, presenting or not the image of a single system. Ubiquitous network computing is a potential user of this kind of Grid because is focused on the tracking of the user in order to allow him to access his computing applications and resources all around the world, adapting those apps to the real resources available (displays, speakers, peripherals, etc) at each moment.

Other concept related to Grid is what is called agent-grids, which are envisioned as the means of making agent-based systems more inter-operable and pervasive. The problem in this case is to find an operational definition of agent, "just because is very difficult to come up with a once-and-for-all definition of agent-hood: one person's 'intelligent agent' is another person's 'smart object'; and today 'smart object' is tomorrow's 'dumb program'." A descriptive definition of an agent may involve a set of attributes which a given agent may have, as: reactivity, autonomy, collaborative behavior, knowledge communication ability, inferential capability, temporal continuity, personality, adaptivity, mobility, etc.

When these kind of agents enter the grid, they receive status information, and their activities are modified and integrated with other activities in the grid, allowing them to update, configure and supply all the information required by their user to or from the network. In this case, the grid is the interconnection path between different agents and allow the interoperability of different kind of agents (including humans and pieces of equipment), providing the ability to easily connect heterogeneous components together to form coherent aggregates. Agent-grids can be seen as a combination of a computational grid and an Agent System architecture, what would mean that it would need to incorporate typical agent system architecture services, like:

- Security, access-lists services, Authorization and Permission,
- Knowledge representation, Transaction services,

- Communication services, Quality of Service network management, Mobility services,
- Monitor and logging services, Charging/Payment services, Environment services,
- Naming/Identification services, Ontology services, Query services, System Management services, etc.

Currently, the most extended kind of grid services are based in computational-grid approaches, usually based on software deployed in networked computers over several operating systems. But all kinds of grids are really merging together -see figure 2- in order to form more powerful systems, where technologies like web services and semantic web concepts, and portable, mobile, multimedia systems are combined, and where huge amounts of data, computational resources and facilities are managed to bring to the user the power of a dream: to have it all in the palm of the hand. In the following sections we'll try to give a summary of the tendencies in this directions.

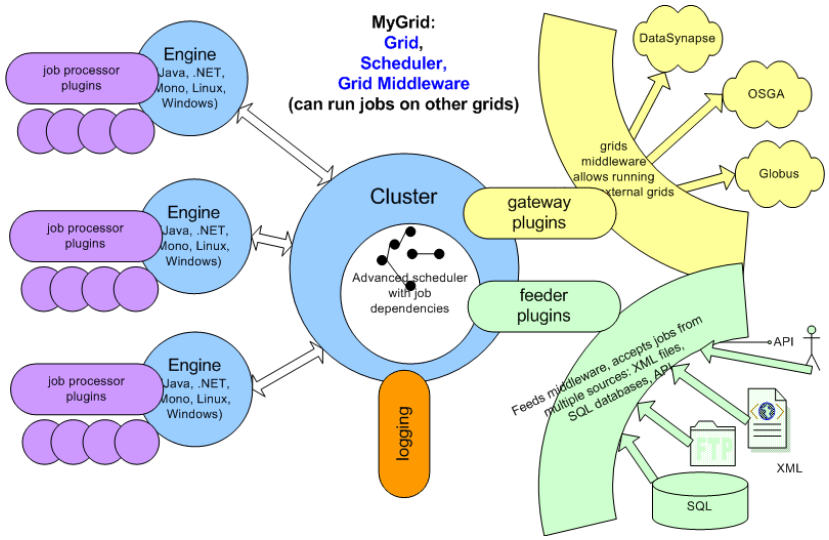

Figure 2: MyGrid, an open-source integrating alternative for Grid deployment

2 Introduction to Portable, Mobile and MM Devices

Is important to realize that our world is full of information, but all this information is not brought to us, the consumers, in only one way but in many and very different: using sounds, images, text, or even all of them combined -what is usually called multimedia (MM)-. This is the starting point of the multiple devices used to do so: letting the user to have the information just when and where it is needed.

Not so many years ago, in the 1980s decade, a portable TV with a 3 inches b/w CRT screen, depending on the car battery was the top on the portable, mobile and MM information technology. Today, not only radios and low power TFT color TVs can be fueled with small batteries, but

many other portable and mobile devices information-related can be mentioned: music players, portable computers, palmtops, cell phones, etc.

Mobility is related to the possibility of accessing the information source in the different locations where the devices can be used, and of course in the way from one place to another. Internet and the big effort done in wireless technologies in the last few years (standards like 802.11b/g or wi-fi, 802.16, bluetooth, satellite, GSM, GPRS and infrared communications among many others digital communication methods) have fueled the expansion of multimedia information to the mobile world.

As portability is power-related, research and evolution of energy storing capabilities and technologies has been very important in many ways: to give devices a better performance by means of new materials for dry batteries, to improve their usability and durability using new voltage regulators and chargers, to allow the use of alternative, sparse and decentralized energy sources as in the case of photovoltaic (PV) solar modules, are some of the most important examples.

Besides of mobility and portability, a new kind of devices was born recently: they are called smart devices and are capable not only to receive, transmit and play all kind of MM information in a portable and mobile manner, but to be a platform where the user can inter-operate and cooperate with different information resources by means of programs, interpreters and smart agents.

A new wave of electronic devices -called wearable, because of their intention of being "weared" by the user all-day-long- have been created in order to be a natural interface for the user, to live longer, to stay more time unplugged, to be smarter and smaller, and to use energy more efficiently while providing the user all kinds of interactivity, information and multimedia communication capabilities.

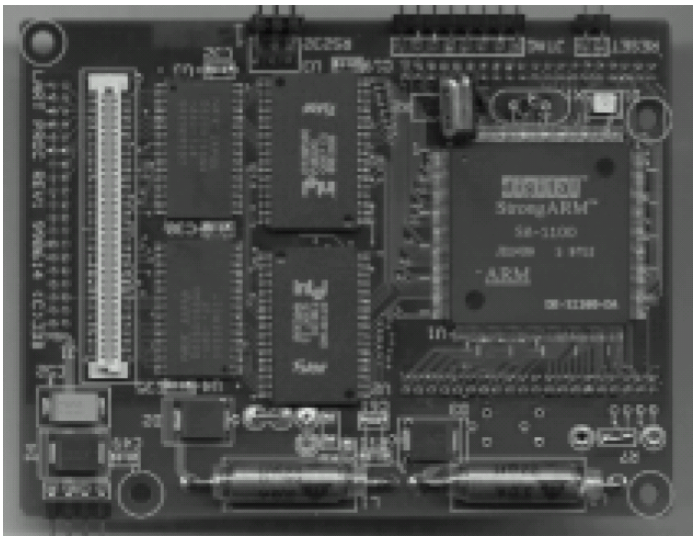

Figure 3: Low-power StrongARM embedded Linux platform

In this sense, a light introduction to dynamic voltage scaling may be very illustrative [2]:

"Power consumption is the limiting factor for the functionality of future wearable devices. Since interactive applications like wireless information access generate bursts of activities, it is important to match the performance of the wearable device accordingly. This paper describes a system with a microprocessor whose speed can be varied (frequency scaling) as well as its input voltage. Voltage scaling is important for reducing power consumption to very low values when operating at low speeds. Measurements show that the energy per instruction at minimal speed (59 MHz) is 1/5 of the energy required at full speed (251 MHz). The frequency and voltage can be scaled dynamically from user space in only 140 μs. This allow power-aware applications to quickly adjust the performance level of the processor whenever the workload changes".

And later on, the same paper in the conclusions section says: "For example, a bursty application like an MPEG decoder can use a model to estimate the processing requirements for each frame. Calculations indicate that with the small delay [...] to set the performance level, the overhead can be less than 1%". See figure 3 for a view on the wearable augmented-reality terminal described, which has a size of 10x7.5 cm, a weight of 50 g, 32 MB of volatile memory, 4 MB of non-volatile memory, a SA-1100 processor, a programmable voltage regulator and various I/O capabilities. The board, called LART, runs under control of the Linux operating system, modified to support frequency and voltage scaling, recalibrating kernel's internal delay routines and adjusting the memory parameters that control the read/write cycles on the external data bus.

3 Mobile MM Services Support with Grid Resources

As stated in [1], the advances in high quality digital wireless networks and differentiated services have enabled the development of mobile multimedia applications that have to execute in global infrastructures, mainly because of (1) their large storage, bandwidth and computation requirements and (2) the limited memory, computation and power on handheld devices. Besides, the mobile nature of the clients leads to frequent tearing down and reestablishment of network connections, introducing latency and causing jitters, which may be unacceptable for delay-sensitive streaming MM applications. Mobile IP techniques may make the latency problem worse, while buffering may be a partial and limited solution to jitter.

The question here is how to exploit effectively the intermittently available idle computing, storage and communication resources in a Grid system in order to achieve real time delivery guarantees in mobile environments. For doing so, two main solutions are the most popular: on one hand, the use of connection-oriented services combined with stream buffering in the path of service; and in the other hand, to trans-code the incoming stream from the server at a local proxy that can customize the MM content based on QoS levels and the resource/power capabilities of the mobile device.

To use locally available idle grid resources to customize MM applications for mobile users based on user requirements and limitations, an effective resource allocation policy must address the performance-quality trade-off. But several complications arise: firstly since grid resources are intermittently available, the policies have to take their availability into account; secondly, the heterogeneity of grid resources and clients complicates the resource management issue; thirdly, user mobility patterns may not be known; and fourthly, MM applications have QoS requirements (e.g. required network transmission bandwidth, accuracy and resolution of displayed images).

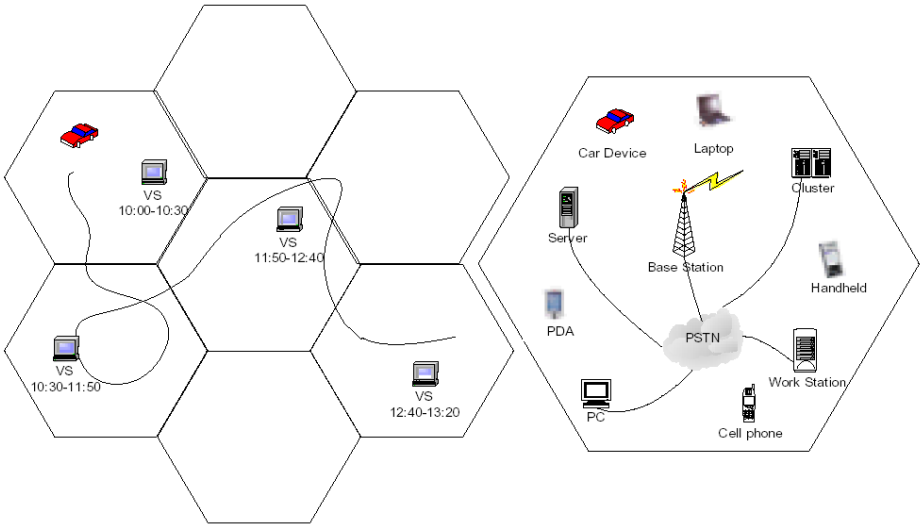

Figure 4: Mobile Grid system environment (left) for providing service for a mobile client by Volunteer Servers (VS) available, (right) possible components within a cell

A suitable middleware framework and the algorithm (called Grid Resource Allocation for Mobile Services, GRAMS) to find Volunteer Servers (VS) for mobile hosts that tries to solve these problems, can be found on [1]. This work not only tries to (a) execute a load-based adaptive grid-server selection or (b) satisfy QoS requirements, but (c) to bound the number of grid-server switches during the lifetime of a request and (d) to adapt to dynamic changes in user mobility and grid-server availabilities. Matters as the degree of location awareness required by the the middleware for efficient allocation of grid resources in a scalable fashion and the trade-offs that arise when timeliness requirements interfere with other application requirements such as security and reliability, still remain opened for further research.

As can be seen, besides of the efforts in the development of sophisticated resource management strategies and algorithms, the efforts for the development of protocols enabling structural negotiation for the use of those resources, are specifically needed. This last, is an area facilitated by formulating and refining the abstraction of Grid services. Based on this abstraction, the Global Grid Forum [11] is developing a set of protocols, called WS-Agreement, for the negotiation, creation and management of agreements for services in the Grid.

Those agreements, as provisioning targets, allow the provider to better estimate demand and therefore make better provisioning decisions, and in last instance can be used to leverage and counteract the changing conditions on the Grid. Three proposals are evaluated by different experiments in [10], but further work is in progress to improve the adaptation strategies and therefore the provider's ability to satisfy the client's QoS. Tendencies go in the way of employing rate limiting to favor under-performing requests whose transfer time is about to end and using multi-path routing in order to adjust the flow rate on different route for QoS satisfaction.

In the grid's dynamic environment, two more issues are of prime importance: performance, related to how quickly the grid system can complete the submitted tasks, and trustworthiness

(that the results obtained are in fact due to the computation requested). This, in a grid system may be worse because the nodes that provide CPU cycles will most likely have computational capabilities that greatly vary between them. The Grid may be composed by any number of high-end nodes and low-end personal computers.

In addition, these nodes are geographically distributed, being possible that a node goes down or becomes inaccessible without notice while it is working on a task. Therefore, a slow node might become the bottleneck of the whole computation if the assembly of the final result must wait for the partial result generated on this slow node. The result may affect performance, going from a small delay to a definitive halt of the whole task if some kind of check-pointing service is not implemented in the grid system.

To most applications, the correctness and accuracy of grid-based computations are vitally important. Faults on the grid servers or nodes may hurt the integrity of a computation. These might include faults arising from the network, the system software or the hardware. Even a user may provide a system to the grid without the intention of faithfully executing the tasks ordered. The resources in a grid system may be so widely distributed that may be very difficult to prevent bad nodes from participating in a grid computation.

As stated in [4], "To enforce the correctness of the computation, many distributed grid systems adapt fault-tolerant methods, like duplicate checking and majority vote. In these approaches, subtasks are duplicated and carried out on different nodes. Erroneous partial results can be found by comparing th partial results of the same subtask executed on different nodes. Duplicated checking requires doubling computations to discover an erroneous partial result. Majority vote requires at least three times more computation to identify an erroneous partial result. Using duplicate checking or majority vote will significantly increase the workload of a grid system".

For Monte Carlo computations (about 50% of the CPU time used on supercomputers at the U.S. Department of Energy National Labs is spent on Monte Carlo computations) some clues on improving performance and improving trustworthiness without increasing the number of CPU cycles needed are proposed in [4].

4 Conclusions

Grid is considered one of the most powerful tools that the computer science has created during its short history. The basic concept of Grid is the possibility of using a set of distributed and networked devices as a unique system, allowing to take advantage of those 90% idle CPU cycles that today computers spill.

However, in order to have a powerful, secure, fast and useful Grid system a big number of problems have to be addressed. Some of them have been introduced in this paper, only in the intention of giving to the reader an overview of the big management complexity of the system itself and of the portable, mobile and multimedia applications necessities.

References

[1] Supporting Mobile Multimedia Services with Intermittently Available Grid Resources, Y. Huang et al., Dept. Information and Computer Science, University of California, Irvine, USA http://mapgrid.ics.uci.edu/HiPC2003.pdf

[2] Dynamic Voltage Scaling on a Low Power Microprocessor, J. Powelse et al., Delft Univ. of Technology, The Netherlands http://www.ubicom.tudelft.nl/docs/UbiCom-TechnicalReport_2000_3.PDF

[3] Characterizing Computer-Related Grid Concepts, F. Manola, Object Services and Consulting, Inc. December 1998 http://www.objs.com/aits/9812-grid-report.html

[4] Grid-based Monte Carlo Application, Y. Li and M. Mascagni, Dpt. Comp. Science, Florida State University http://www.cs.fsu.edu/~mascagni/papers/RICP2002_3.pdf

[5] GPU, a framework for distributed computing over Gnutella, T. Mengotti, ETH Zürich, March 2004http://gpu.sourceforge.net/gpu_p2p/

[6] MyGrid Project http://mygrid.sourceforge.net

[7] GPU Project http://gpu.sourceforge.net

[8] European DataGrid Project http://web.datagrid.cnr.it

[9] The Globus Alliance http://www.globus.org

[10] Providing Data Transfer with QoS as Agreement-Based Service, H. Zhang, K. Keahey, W. Allcock. 2004. http://www.globus.org/research/papers/1568935481_zhang_h.pdf

[11] Global Grid Forum http://www.gridforum.org