Virtual Application Environments on Computing Grids

Toan Nguyen, Lizhe Wang
Institut National de Recherche en Informatique et Automatique
F-38334 Saint-Ismier Cedex, France
e-mail: ***Toan.Nguyen@inrialpes.fr***
http://www.inrialpes.fr/opale

and

Jean-Pierre Antikidis
Centre National d'Etudes Spatiales, 18, Av. Edouard Belin
F-31401 Toulouse Cedex 9, France

ABSTRACT

Because new usage are expected from upcoming telecommunication and computer technologies, new applications that are orders of magnitude larger than current single discipline applications are likely to be addressed in the near future. Therefore, new computing technologies and practice are also required.

Among these technologies are parallel and distributed computing, in cluster and grid-based environments. It is clear that large PC-clusters and wide area grids are currently used for demanding scientific applications, e.g., nuclear and environmental simulation. It is not so clear however what kind of new business models they will effectively support.

We advocate in this paper the use of a grid-based infrastructure that is designed for a seamless approach to the new e-business users, although it relies on sophisticated computing environments based on computing grids, i.e., wide-area computing grids, connecting heterogeneous computing resources: mainframes, PC-clusters and workstations running application codes and utility software, e.g., remote processing and visualization tools.

The approach is based on concepts defined by the HEAVEN* consortium. It is a European task force that includes industrial partners from the aerospace, telecommunication and software industries, as well as academic research institutes. The goal is to define, develop and provide test-beds for emerging applications and business in the forthcoming Information Society and to explore new usage of computing technologies in the economy and industry during the next decades.

Key-words: virtual environments, distributed applications, emulation, grid computing

1. INTRODUCTION

The HEAVEN consortium works on a project aiming at the creation of advanced services platforms supporting applications in research, science, business and community services. Along the line of the strategic objective "GRID-based systems for complex problem solving", it is an R&D project in the field of "Enabling Application Technologies" based on grid infrastructures. It is intended to enable "virtual application environments", a kind of "virtual private grids", which support various configurations for users using a suitable high-level description language. This will become the basis for future generalized services allowing the integration of various services without the need to deploy specific grid infrastructures.

The users can define their own "virtual" computing environments by selecting the appropriate communication and computing resources required, or reuse existing environments. The approach is generic by allowing various application domains to benefit from potential hardware and software resources located on remote computing facilities in a simple and intuitive way. Basically, the user interface provides an icon-based request facility that allows to dynamically define the computing environment best-suited to the end-users needs.

The computing resources are defined by services available as sets of standardized interfaces performing specific tasks: application workflow, input data streams, output visualization tools, monitoring facilities, telecommunication networks, etc. Services can be composed and hierarchically defined. Transparent access to heterogeneous hardware and software operating systems is guaranteed.

*HEAVEN ("Hosting European Applications in Virtual Environments") is a collaborative project involving the project OPALE at INRIA (France), the European Aeronautic Defence and Space Company (EADS) Corporate Research Center (France), Centre National d'Etudes Spatiales (CNES) Program and Strategy Directorate, Space Information Systems (France), DATAMAT (Italy), SciSys (Great-Britain), IDEC (Greece), the University of Paris 6 (France), the University of Cyprus at Nicosia and other pending candidates.

Based on current and expanding grid technology, various approaches have been designed to support the deployment of scientific and business applications on distributed networks of computing resources. High-performance computing has played a major role in this area, when large scientific computing applications have been design on shared clusters of remote computers in the late eighties [ASCI, y, z]. They led to the design and dissemination of middleware supporting application deployment, data sharing and transfer and task synchronization in a standardized form [Globus]. Further, various dedicated environments have been designed to support scientific application deployment on grids [Cactus, Legion, Condor].

Because these approaches are mainly based on a compromise between the "application push" and "technology pull" paradigms, they encounter specific problem when disseminated among the user communities. The major drawbacks concern the ease of use, the best usage practice recommendations (or lack thereof), the maintenance and set-up burdens and the accounting policies available. All these items require adequate expertises which restrict them to the dissemination to particular sets of user communities, i.e. those that can afford the support of proper maintenance, experts and technical staff.

The advent of Beowulf clusters [beo] and cheap aggregation of commodity and off-the-shelf computers forming large computer farms raises however new challenges. One is the set-up and maintenance of adequately corresponding cheap middleware. The other is the development of new business models based on these new, large and powerful computing environments.

This paper presents an approach for the design, development, deployment and execution of large distributed applications on virtual environments based on computing grid technology. Because the current technology already advocates virtualization as a free lunch when using grids, it is necessary to explain what "virtual application environments" are. This is the subject of Section 2. The technicalities underlying this approach are detailed in Section 3. The benefit of using this approach is detailed in Section 4. Section 5 is a conclusion.

2. VIRTUAL APPLICATION ENVIRONMENTS

Goals

Virtual application environments are tools and facilities dedicated to the design, deployment execution, monitoring and maintenance of large applications on distributed resources. These resources may be computers, file archives, sensors, visualization environments, etc. The users do not need to own any one of them. He or she may have access to and use any combination of them among a set of available resources whenever he or she is granted the appropriate rights to do so, using a simple laptop of sophisticated apparatus, e.g., an immersive CAVE [CAVE].

He does not need any technical knowledge of the underlying software and hardware tools, except that one he or she is currently using. The technical infrastructure, may it be a state-of-the-art middleware for grid computing of a large cluster of commodity PC connected through a high-speed fiber-optics network is made totally transparent to him/her.

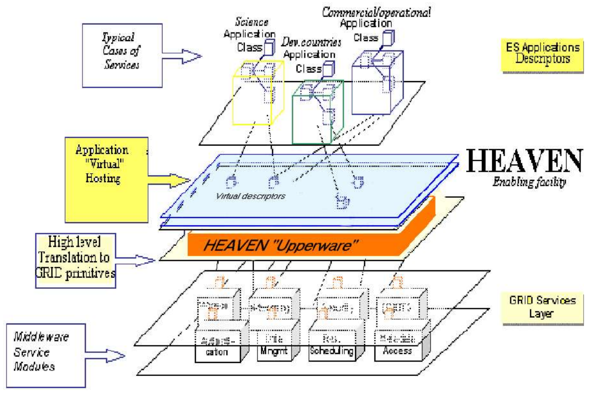
Typical Cases of Services
Science Application Class
Dev.countries Application Class
Commercial/operational Application Class
ES Applications Descriptors
Application "Virtual" Hosting
Virtual descriptors
HEAVEN
Enabling facility
High level Translation to GRID primitives
HEAVEN "Upperware"
GRID Services Layer
Middleware Service Modules
cation
Mngmt
Scheduling
Access

Figure 1. The HEAVEN approach.

The users can request the allocation of specific software and hardware for running their applications, ranging from satellite data acquisition antennas and processing software, to supercomputing mainframe with the corresponding QoS and schedule.

Stated otherwise, the virtual application environments designed here are dedicated to the end-users and the application designers. They are not intended to be used as working environments by maintenance staff or engineers and experts that need more sophisticated technical information than the user communities (task scheduling, load balancing, authentication and authorization, etc).

Background

The idea implemented here is that current middleware, although they advocate virtualization as a natural and free side-effect of their services and facilities, do not provide the target communities with easy-to-use, simple and affordable virtual application environments. They require steep learning curves and a staff of expert to fulfil operational requirements. This is not realistic from an industrial point of view in current business and industrial competitive environments and markets. For this to become a daily reality, it is necessary to provide virtual application environments where the application designers and the users will not have to become middleware experts. This can be achieved in a way similar to the Internet development history: by providing powerful though sophisticated application development environments, in a way similar to the web site development kits, web browsers and web page editors, which do not even require the knowledge of the html languages and variations.

This requires filling the gap between the user and application developers communities and the existing middleware. This is done here by the development of a software layer which masks to these communities the idiosyncrasies of grid computing, task scheduling, resource brokering, load balancing, QoS, etc. We call this software layer the “upperware”. Its goal is to simplify the definition, deployment, monitoring and execution of the applications by masking the various concepts implied with a unique self-described “virtual resource” concept. A satellite tracking sensor, a PC-cluster, a data file and a post-processing task are all “virtual resources”. They may be implemented by various means on different computing environments, e.g., Web services, distributed components, or even standardized APIs. This only depends on the local processing environment capabilities, and there is no monolithic model compliance requirement. This is necessary for the support and scaling of heterogeneous systems and devices.

Because the goal is to simplify the end-users and application designers life, the sophistication of the upperware has to take into account the underlying middleware concepts, e.g., resource brokering and reservations, QoS, accounting, logging, without user interaction to the best possible extent.

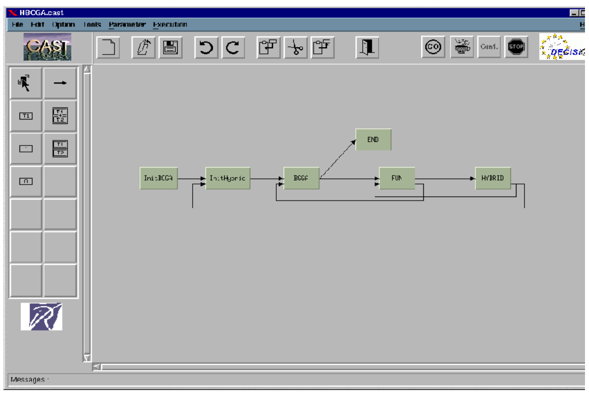

Figure 2. HEAVEN: the user interface.

Further, the upperware must be able to emulate full-fledged application environments corresponding to the applications requirements, even when the end-users or their companies do not own any one of the particular resource involved. This means that they may invoke commercial and outsourced resource and data centres that will charge for the resource usage. In this case, the upperware is the unique interface between the users and the various systems providing the resources. These may be transparently invoked provided the upperware implements adequate remote access mechanisms.

3. TECHNICALITIES

User aspects

From the user point-of-view, the interface with the virtual application environment is a high-level graphic interface that masks the resource distribution and technical definitions. It is a set of dependent tasks connected by a workflow graph (Figure 2). This approach leaves all the technical aspects to a further step, while focusing on the application logic only. The tasks can be connected by a control flow graph formed by sequence, parallel, interleaved and imbedded loops.

The tasks correspond to executable codes that are located transparently for the users on remote sites. It is the responsibility of the application designers to define which resources the application needs, where they should be located if required, and which complementary properties they should exhibit (availability, QoS, etc). None of these resources are required to be local and to belong to the users and designers. Brokering protocols and usage grants are therefore supported by the upperware. Submission of such grants can be negotiated on a permanent or one shot policy. The upperware appears therefore as a general resource broker, negotiating with the remote systems the availability and usage of resources, based on the local policies and granted access rights.

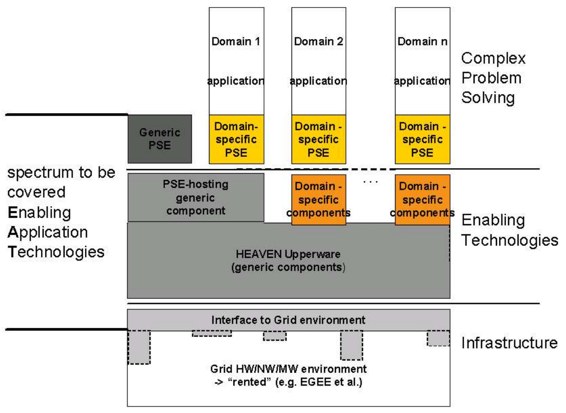
Domain 1 application
Domain 2 application
Domain n application
Complex Problem Solving
Generic PSE
Domain-specific PSE
Domain - specific PSE
Domain - specific PSE
spectrum to be covered
Enabling Application Technologies
PSE-hosting generic component
Domain - specific components
Domain - specific components
Enabling Technologies
HEAVEN Upperware (generic components)
Interface to Grid environment
Infrastructure
Grid HW/NW/MW environment -> “rented” (e.g. EGEE et al.)

Figure 3. The HEAVEN architecture.

Technical aspects

From a technical point-of-view, the upperware is a software layer that is based on existing grid infrastructures, e.g., EGEE, RENATER, etc. As such, it interfaces both the user communities through the high-level grahic interfaces described above, and the underlying computing environments. It fills the gap between them and the application problem-solving environments (Figure 3). It includes generic components for interface with grids (invocation and negotiation with remote resource brokers, authentication and authorization grants negotiations, etc). It also supports specific components dedicated to particular application requirements (interface with sensor management systems, with visualization tools, etc). Finally, it is the basis on which the particular application domains solve problems.

There are several ways to implement the upperware, for example by a generic Web services implementation [WSDL] and a component-based architecture [CCM]. The first option is preferable since it guarantees the compatibility with the existing OGSA architecture [OGSA] and the hopefully soon widely used Globus GT4 [GT4]. Further, compatibility with forthcoming versions of other middleware such as UNICORE [Unicore] which are now interoperable with Globus [GRIP] will be supported. There is however no guarantee that a backward compatibility with previous versions of Globus Toolkits (GT2 and GT3) will even be supported by GGF

[GGF]. Therefore, this is not a priority concern for HEAVEN.

4. BENEFITS

Abstraction

The obvious advantage of the virtual application enviroments are their ability to mask the technical aspects of grid technology to the application designers and users. The example depicted by Figure 1 is an aerodynamics optimization application running on three remote PC-clusters located in different locations at INRIA centers and connected by a high-speed gigabits/sec network (Figure 5). The end-users never interact directly with the underlying middleware and network. The application designers have to define the abstract tasks involved, the corresponding executable codes (by their name and access paths) and the resulting data files (by their names and access paths also).

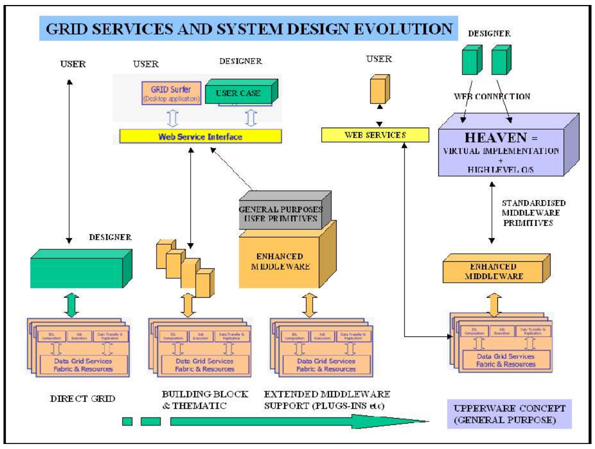
GRID SERVICES AND SYSTEM DESIGN EVOLUTION
DESIGNER
USER
USER
DESIGNER
USER
GRID Surfer (Desktop application)
USER CASE
WEB CONNECTION
Web Service Interface
WEB SERVICES
HEAVEN = VIRTUAL IMPLEMENTATION + HIGH LEVEL O/S
GENERAL PURPOSES USER PRIMITIVES
STANDARDISED MIDDLEWARE PRIMITIVES
DESIGNER
ENHANCED MIDDLEWARE
ENHANCED MIDDLEWARE
Data Grid Services Fabric & Resources
DIRECT GRID
BUILDING BLOCK & THEMATIC
EXTENDED MIDDLEWARE SUPPORT (PLUGS-INS etc)
UPPERWARE CONCEPT (GENERAL PURPOSE)

Figure 4. The HEAVEN functional components

The workload is here only to define the matching inpu and output parameters of the various abstract tasks, which, in the case of large scientific computing applications, can be in very large number. The application communities therefore only focus on the application logic (Figure 4).

The grid technical aspects are reduced to a minimum. The IP addresses of the various front-ends to the local PC-clusters must be given, as well as the name of the local services used as local proxies to the executable codes.

This leaves the authentication, authorization, resource brokering and QoS negotiations where they should be : on the middleware side. The users never interact with them. The grid and underlying networks are therefore made transparent. This is similar to the Internet protocols and their underlying networks, which are totally masked to the casual users.

Scalability

The second benefit, besides grid transparency, is the flexibility and scalability of the virtual application environments. Should the user want to change any of the executable code corresponding to an application task, he/she can change it by updating the access path and name of the corresponding code, without modifying the application environment at all.

In a similar way, should the resources be changed, because of failure or maintenance considerations, this can be implemented transparently for the user communities.

This also supports seamlessly the scalability of the environment itself, of the grids and of the resources involved. It does not impact existing application definitions and user interactions with them.

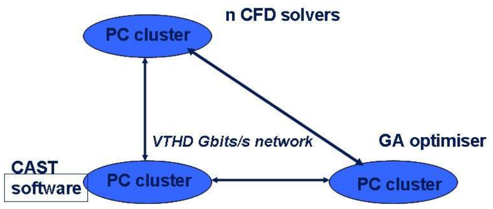

Figure 5. The HEAVEN deployment.

A first prototype called CAST [4] was implemented using the ideas above. It has been tested on aerodynamics optimization test-cases. It did not include interfaces with existing grid middleware, but rather used an extended CORBA layer for communication and synchronization. This layer handled MPI parallel codes as single CORBA objects. This abstraction layer also supported the specification, deployment, monitoring and execution of distributed simulation test-cases using both CORBA and non-CORBA compliant routines.

A clear client-server decomposition of the application was made possible by CORBA (Figure 6). The application specification in the CAST software is a client to the optimization process, which is itself a client for the particular optimization routines used. All these tasks are located remotely on different PC-clusters running the corresponding servers (Figure 5).

An interesting side-effect of running the codes on PC-clusters is that the codes can use parallelization techniques. Indeed, domain decomposition techniques are used for the CFD codes, which benefit from the large number of processors available on each particular cluster. This greatly improves performance figures of the CFD code, and consequently, of the overall optimization application.

Integration

A third benefit is that, while adequate for the application development and maintenance, this architecture lends itself nicely to interfaces with grid middleware like UNICORE also. Such an implementation is currently being developed by project OPALE [opale] at INRIA.

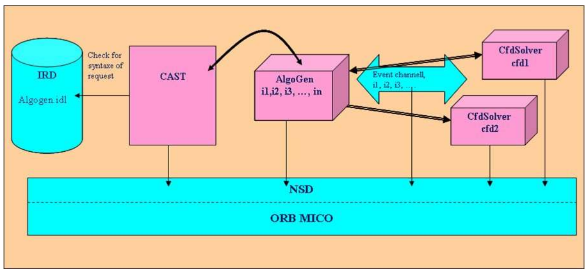

Figure 6. The CAST architecture.

Further, an interesting ability for UNICORE is to support application definitions using the dataflow and a limited form of workflow approach. This is complementary to the CAST approach which adopts a workflow approach based on SCCS [sccs]. The combination of both approaches, while extending the facilities to deploy composite application, also support the existing heterogeneity of current software development which must take into account legacy applications.

Outsourcing

A fourth benefit is that application designers can define their own computerized environment, including sensors, databases, visualization and post-processing tools together with applications code. They do not have to depend on existing in-house hardware and operating systems because the HEAVEN upperware masks them in the underlying middleware layer. This is truly the outstanding benefit of the virtualization approach: the application environment can be defined without owning any of its specific components. The latter can all be rented, used, and paid for at commercial computer data, telecom and processing centers. A nice side-effect is also that the application designer can run his/her proprietary application code on these "outsourced" virtual application environments, as usual.

Business models

Last but not least, another beneficial side-effect is here that new business models are emerging using this approach. Data resulting from the application executions in their virtual environments can be charged for their added-value without ever accessing the original files by the end-users, and without owning the processing environments by the data provider. This scenario is currently being designed for spatial data marketing by CNES [Hangzhou].

5. CONCLUSION

The exponential growth of distributed and cluster computing on wide-area grids is an obvious challenge for computer scientists and all the communities of users today. If grid-computing is to break the casual-users barrier, like the Internet did ten years ago, many challenges remain to be addressed. One of the fuzziest and creeping challenge is the ease of use and best-practice standards for grids. There are still no clear tools and methodologies answering these questions today. Ease of use will clearly convince reluctant user communities from the scientific, industry and business arena to adopt this promising technology. One approach is to devise new interfaces to grids that will help the users to abstract their applications from the technicalities of the underlying and ever-growing technologies supporting the computerized world.

This paper presents a new paradigm based on the full virtualization of resources involved in the applications. It abstracts all the involved resources in a technology independent upperware. This is a software layer that builds on existing grid middleware, taking benefit from the Web Services technology to build transparently a standard abstraction layer masking the underlying grid infrastructures. We call it "Virtual application environments" because there is no need to own any of the resources involved. Consequently, it therefore paves the way to new business models. It is in no way another grid middleware. It is instead a software layer masking the intricacies and technical details that no user community can today fully understand, deploy and maintain without the help of dedicated teams of professional experts.

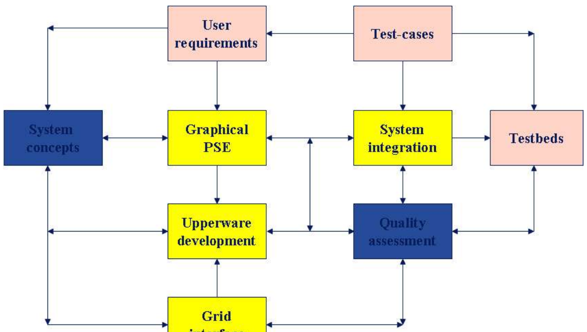

Figure 7. The HEAVEN functional breakdown

It is our belief that no other solution exists today for the wide dissemination of the very promising grid technology. The recent world-wide expansion of the Internet is the best analogy we can find and a convincing contributing proof to this approach: only a very little percentage of today's Internet users really understand the underlying technology. But its usage is an everlasting world-wide expansion.

Our thesis is that, by far, this is not the case for grid technology. It is still mainly used by highly skilled professionals. The biggest success stories of grid technology still remain ahead of us, when most of its users will have only very little knowledge of what is going on inside. HEAVEN is a small contribution in this direction.

6. ACKNOWLEDGMENTS

The authors wish to thank the partners of the HEAVEN consortium for their strong support and their participation to many invaluable discussions concerning the HEAVEN upperware. They also particularly appreciate the support of Guillaume Alleon and Eric Duceau from EADS Corporate Research Center for their advice and expertise.

7. REFERENCES

[1] Nguyen G.T., J.P. Antikidis. Virtual computing environments for problem-solving on grids. International Parallel CFD Conference. Universita de Las Palmas. Gran Canaria (Spain). May 2004.

[2] Nguyen G.T. Grid-computing in Multidisciplinary CFD optimization problems. International Parallel CFD Conference. Invited lecture. Russian Academy of Sciences. Institute of Mathematical Modeling. Moscow (Russia). May 2003.

[3] Nguyen G.T. Integration of multidiscipline applications in grid-computing environments. PARA02 Conference on Applied Parallel Computing. Center for Scientific Computing. Helsinki (Finland). June 2002.

[4] Nguyen G.T. An Integration platform for metacomputing applications. International Conference on Computational Science ICCS 2002. Amsterdam (NL). April 2002.

[5] Nguyen G.T., L. Wang, J.P. Antikidis. Virtual environments for spatial data infrastructures on computing grids. Paper submitted for publication. April 2005.

[4]GGF
[4]OGSA
[4] G.T3
[4] G.T4
[4] G.T2
[4]GRIP
[4]WSDL
[4]CCM
[4]Globus
[4]CAVE
[4]Condor
[4]Legion
[4]Cactus

e

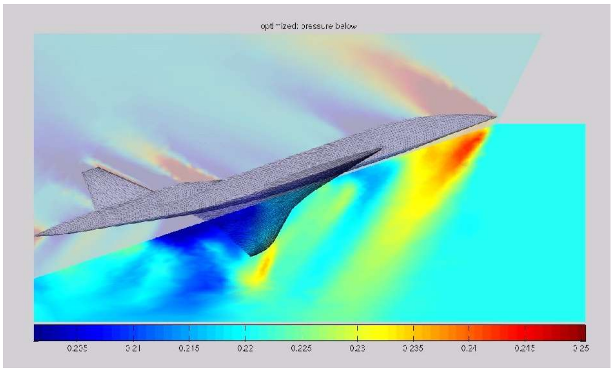

Figure 8. A supersonic wing optimization by OPALE project at INRIA.

8. WARNING

This is the draft of a paper accepted for presentation at the Third International Conference on Computing, Communications and Control Technologies (CCCT'05), Austin, Texas (USA), July 24-27, 2005.
Please, do not copy.