ProActive User Group

and

Grid Plugtests

October 18-20th 2004

Authors: OASIS Team and ETSI

1. Overall Objectives

ETSI and INRIA organised a three-day event, which started on October 18th 2004. The objective was to learn, through the ProActive user experience and through open discussion, about the future features needed for the Grid middleware as well as to get important feedback on the deployment and interoperability of Grid applications on various Grid platforms.

The event consisted of three different happenings. On the first day, talks were given regarding the use of ProActive, from introduction of the middleware, to descriptions of its use in the industry, and it's current research status. The second day was dedicated to a contest between 6 teams, where the aim was to find the number of solutions to the N-queens problem, N being as big as possible, in a limited amount of time. The last day allowed the final discussions and prizes where distributed to the winners.

This event was organised under the supervision of UNSA and I3S CNRS, and was sponsored by IBM, SUN and ObjectWeb.

2. ProActive & User Group

On the three days, the event drew 80 participants, from 10 different countries : France, Chile, USA, England, Holland, Switzerland, Spain, Italy, Japan and Korea. All these people met to share their views of ProActive, the Grid middleware developed in the OASIS team, INRIA.

2.1 ProActive

ProActive is an LGPL Java library for parallel, distributed, and concurrent computing, also featuring mobility and security in a uniform framework. With a reduced set of simple primitives, ProActive provides a comprehensive API allowing to simplify the programming of applications that are distributed on Local Area Network (LAN), on clusters of workstations, or on Internet Grids.

The deployment descriptors provide a mean to abstract from the source code of the application any reference to software or hardware configuration. It also provides an integrated mechanism to specify external processes that must be launched and the way to do it. The goal is to be able to deploy an application anywhere without having to change the source code, all the necessary information being stored in an XML descriptor file.

Since programming the Grid cannot be achieved at a low-level of abstraction, ProActive is provided with a programming model. The complexity that arises from scale, heterogeneity, and dynamicity cannot be tackled with message-level primitives. As such, development of new Grid programming models has to rely on higher-level of abstraction than the current usage. These programming models are based on the component technology.

ProActive talks were held on the first day. In the morning, the overall frame the middleware allows was presented, with talks underlining it's main aspects. On the afternoon session, users were invited to speak about their use of the middleware. During the evening, future work was presented, and a panel of experts was invited to talk about actual problems in the Grid domain.

2.2 ProActive technology presentations:

On Monday morning, an overview of what ProActive has to offer was given. After a welcome speech, a first session covered the basic programming features, and a second session went into more detail into composing and deploying applications using ProActive.

9:15 **“Welcome”, Karl-Heinz Rosenbrock, ETSI DG and P. R. Guillemin,**

Session 1: "Basic Programming Features" 9.30-11.00

9.30 **"ProActive Overview", D. Caromel, UNSA**

ProActive aims at providing a complete solution for the Grid:

o programming, o wrapping,
o composing, o deploying.

Based on an Active Object model, asynchronous methods calls with first class futures is at the root of the programming model. Wait-By-Necessity enforces that futures are implicit, and can be passed as parameter and results between distributed machines. The programming model enjoys a strong property of determinism. Finally, put together with features to be presented in next talks, the 4 key elements:

o asynchrony, o wait-by-necessity,
o groups, o components,

should be the essence of a promissing Grid alchemy.

10.00 **"Group Communications, and OO SPMD", L.Baduel, UNSA**

The group communication mechanism of ProActive efficiently achieves asynchronous remote invocation for a group of local or remote objects, with automatic gathering of replies. Given a Java class, one can initiate group communication using the standard public methods of the class together with the classical dot notation; in that way, group communication remains typed.
The group communication mechanism, joint to a small sized API, allows the programmer to build Object-Oriented SPMD applications, using data flow synchronization.

10.30 **"Mobility", F.Huet, UNSA**

ProActive provides mechanism to perform weak migration of any serializable active object. The basic API consists in a single migrateTo method. We also provide a high level API to build itineraries that can be followed by mobile objects to perform various tasks. In order to maintain communications, we use two schemes. The first one relies on forwarders left on each site visited by an agent, while the second uses a centralized server. A third one, using both forwarding and a server, is currently under development.

Session 2: "Composing and deploying" 11.30-13.00

11.30 **"Components and Legacy Code", M. Morel, INRIA**

The aim of this presentation was to show the relevance of component based programming, and particularly the Fractal component model, in the context of Grid computing. ProActive components benefit from all standard features of the ProActive library, and future works will focus on legacy code wrapping, automatic data redistribution between components and dynamic optimizations.

12.00 **"ProActive Grid Deployment and GUI", Romain Quilici, UNSA**

This talk covered ProActive's deployment infrastructure, and presented also ProActive's GUI that provides monitoring and control over deployed Active Objects. In the context of the GRID, where main concerns include scalability (large number of machines), heterogeneous resources (OS, security, model,...). ProActive provides a flexible and scalable deployment infrastructure based on XML files that provides deployment on various model of Grid. Indeed ProActive is interfaced with almost all Grid protocols: SSH, GSISSH, GLOBUS, LSF, PBS, SGE, PRUN, OAR, RSH, RLOGIN (for Desktop Grid)and also allows combinations of those protocols. Hence we were able using such features to build an heterogeneous Grid all over the world , and to use that Grid for the Plugtests and contest.

ProActive provides also a GUI called IC2D to monitor, visualize and control deployed Active Objects, it offers usefull graphical and textual informations, about the AO's state and allows control such as migration, termination of the JVMs, Nodes, AOs.

12.30 **"Security", A.Contes, INRIA**

As ProActive applications could be transparently deployed in many ways, security related-code should not be tied to source code, but rather to deployment files. The security model involves hierarchical domains, dynamic policy negotiation, extensible security policies and deals with activity migration. As a perspective, we want the security model to interact with all new ProActive features like components and fault tolerance mechanism.

2.3 User presentations:

After lunch, some people were offered the opportunity to show how ProActive can be used to its full potential. The talks showed various applications, and stressed the versatility of ProActive, which is capable of providing its services to many different applications, thanks to its generic programming model.

14.30 **"Distributed agent based simulations with ProActive"**,
Luc Girardin, Swiss Federal Institute of Technology
(ETH) and Macrofocus Zurich, Switzerland

We are interested in understanding the role played by macro-historical processes in the emergence of political violence, such as state-formation, nationalism, and democratization. To trace such transformations, we rely on a agent-based modeling approach and use a

Java library, RePast, specifically targeted at this type of simulations. To speed up the computation of our simulations, we explain how RePast was extended to support distributed simulation runs using ProActive, and discuss how a more elaborate parallelization scheme can achieved through a double-buffering technique. ProActive has enabled us to not just improve our simulations from quantitative point of view, but also qualitatively as it let us get results almost interactively instead of having to wait overnight.

15.00 **"ProActive based Architecture and Application for the management of Electrical Grids"**, E.Zimeo. RCOST, Italy

The talk discusses the exploitation of *ProActive* for the definition of a component in a distributed Web-based architecture for the security analysis of electrical Grids. The architecture is characterized by a network of sensors for acquiring data from the components of an electrical network, a computational engine based on ProActive for processing data coming from the sensors and a data base for storing the results of the computation, used for performing statistical analyses. The computational engine is able to execute a concurrent algorithm to solve the power flow equations for each possible contingency (fault) in the electrical network. The algorithm is transparently executed by using a hierarchical master slave model mapped on a hierarchical network of computational resources through the distributed implementation of the hierarchical master/slave design pattern. The allocation of each slave task to a different computational resource belonging to a set of heterogeneous resources is improved and automated through the adoption of a broker provided by HiMM (*Hierarchical Metacomputing Middleware*), a middleware for Grid computing developed at the University of Sannio, on top of which a specific implementation of ProActive was carried out. By using a variant of the time minimisation algorithm proposed by Buyya, the broker is able to select in a network of heterogeneous computers the ones that guarantee the termination of the security analysis in a prefixed time, specified by the user as desired QoS for the computation. Moreover, the talk illustrates future enhancements of the computational engine through the use of ProActive groups and some their enhancements. Groups make easier and more efficient the implementation of the hierarchical master/slave pattern on a hierarchical physical network. However, to achieve this goal, ProActive groups have to be extended both for the semantics and the implementation. The semantics could be extended by introducing the possibility to specify (at creation time or later) the policy adopted to distribute the invocation of a method to the internal replicas. On the other hand, the efficiency could be improved by implementing the groups on reliable multicast protocols. The availability of such kind of groups will make possible an effective and efficient framework based on ProActive for hierarchical master/slave computing on the Internet.

15.30 **"Computational Electromagnetism on the Grid: a ProActive time domain finite volume solver for 3D Maxwell Equations"**, S. Lanteri, CERMICS/INRIA/UNSA

We discuss the application of distributed objet oriented programming in Java to the development of a grid aware tool for the numerical simulation of large-scale, 3D, electromagnetic wave propagation problems. The time domain Maxwell equations are solved using a finite volume solver on unstructured tetrahedral meshes. The ProActive library greatly facilitates the development of a distributed object oriented SPMD version of the solver. Preliminary performance results are presented for calculations that have been performed on a cluster of PCs.

16.00 **"Distributed BLAST with ProActive" Santosh Anand, Institute of Signaling, Developmental Biology and Cancer, Nice**

Protein and DNA sequence comparison is one of the most important tool of molecular biologists, but sequence databases are growing at an exponential rate, and sequence comparison is becoming increasingly computationally intensive. We propose to develop a GRID-enabled parallel blast, GeB, which is implemented on the ProActive platform. With some very good group communication facilities provided by ProActive, we achieved good speedup and scaling results (up to 39 processors). Encouraged by this, we next move to work on a better load-balancing facility which might give a very good speedup for our GeB.

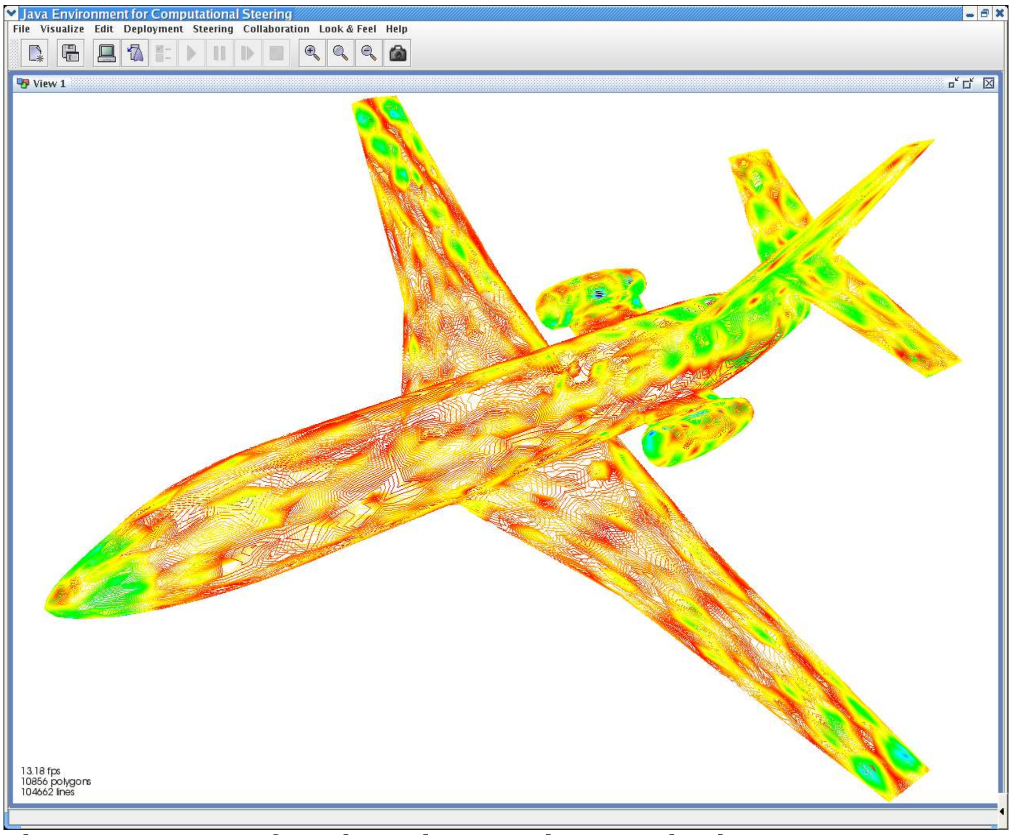
17.00 **"Evaluation of ProActive and other middlewares for a Data Driven Environement for Multiphysics Application" John G. Michopoulos, U.S. Naval Research Laboratory, Washington DC, USA**

The recently experienced enormous progress and proliferation of distributed computing, has lead to the development of a very large amount of middleware technologies. The efforts are originating from groups distributed worldwide and span various areas of interest. The pluralism of particular architectures and implementations along with the custom and generalized needs emanating from the need to develop particular applications, force the idea of a comparative study of the various middleware platforms.

In an effort to select a middleware platform for developing an architecture and an implementation of a data driven environment for multiphysics applications (DDEMA) we were forced to develop a comparative strategy of the available technologies and subsequently apply it for evaluation purposes. According to this strategy we utilized DDEMA’s design requirements to help us define a two stages approach. In the first stage a set of knock out (accept-reject) criteria were developed and applied to all identified middleware platforms. In the second stage a second set of criteria were developed and applied to the platforms that survived the first stage in a manner that allowed assignment of weighted ranking.

This approach enabled the quantitative comparison of “ProActive” (a platform of main interest because of its maturity) with other agent middleware platforms. The results clearly demonstrated that the higher performers were of ProActive, Jade/Leap and Grasshopper. However, it was established that developer familiarity to a particular platform can often overtake any other performance metric.

During the breaks, there was a live demo in the hall: "JECS: Steering and Visualization of 3D numerical simulations on the Grid", Said El kasmi, S. Lanteri, et al. This proved how ProActive can use two different machines collaboratively.


Java Environment for Computational Steering
File Visualize Edit Deployment Steering Collaboration Look & Feel Help
View 1
13.18 fps
10856 polygons
104662 lines
Electromagnetism of a Falcon plane, as shown in the demo

2.4 ProActive perspectives

In the later part of the afternoon, some work-in-progress was presented. This session was meant to show what future features are expected to be offered by the middleware, and what the burning topics are today in the Grid computing field.

17:30 **“Web Services, C# .Net interop, HTTP, SSH Tunneling” Virginie Legrand**

ProActive applications separated by firewalls may need to communicate. The issue is that using RMI communication, more than one port must be opened on these firewalls. We improved a new communication layer based on HTTP in order to use only the usual opened port. Users can also monitor ProActive applications from any web services enabled languages thanks to the ability to export an active object as a web service.

17:40 **“Fault Tolerance” Christian Delbé**

Next release of ProActive will include fully-transparent fault-tolerance feature. Based on a communication-induced checkpointing mechanism, this feature allows ProActive applications to restart automatically after the failure of one (or more) of the active objects, while preserving the work done until this failure. There is no need to modify nor recompile an application to make it fault tolerant.
This work first targets applications running on a single cluster ; we

are currently designing a global protocol based on this work for large applications running on the Grid.

17:50 **“Peer-To-Peer computing” Alexandre di Costanzo**

We aim to use sparse CPU cycles from organizations desktop workstations. First, we are proposing a P2P infrastructure to create a network of JVMs (computational nodes); this infrastructure is self-organized and tunable. The second part is to provide a high level model of P2P programming by an API to solve Branch and X problems.

18:00 **“Dynamic Load Balancing of Active Objects” Javier Bustos**

The goal of this study is to show the potential of ProActive, developping a load balancing architecture only using the API of ProActive. This load balancing is based in two principles: each node will only know its own load (reduces the network load) and each node will make its own decisions (avoids the use of a central load balancer, which is not scalable in practice). These decisions have to follow two objectives: each machine wants to work all the time and the application wants to be optimal. For the study of this problem, we divided the decision in five questions: "How to coordinate the Load Balance?", "Where to migrate active objects?", "When to initiate the migration of active objects?", "How many Active Objects should we migrate?" and "Which active object has to migrate?". Experimentaly we found the answer for the first two questions and we are studying the third.

18:10 **“OSGi” Françoise Baude**

We plan to run ProActive applications as OSGi bundles. The obvious advantage comes from the fact that the OSGi Alliance
Service Platform (www.osgi.org) adoption is very large because it enables services for networked devices (e.g. in industrial or home automation, vehicles, smart phones, etc). One ProActive application we foresee is a remote management tool of OSGi services and platforms.

18:20 **“Formal Verification and Behavioral Properties” E. Madelaine**

We build tools for extracting automatically behavioral models from the source code of ProActive applications, and use model checking techniques to prove their temporal properties (dead-lock freedom, reachability, liveness). This will also lead to techniques for proving correct the assembly of distributed components.

2.5 Panel: Stateful vs. Stateless Web Services for the Grid

The day was concluded by a panel involving experts, answering current questions on the Grid challenges.

18.30 **Panel: "Stateful vs. Stateless Web Services for the Grid: how to get both scalability and interoperability?"** Denis Caromel (UNSA), Tony Kay (SUN Microsystems), Jean-Pierre Prost (IBM EMEA Grid Computing) , Vladimir Getov (University of Westminster), Marco Danelutto (University of Pisa), Christophe Ney (ObjectWeb).

Questions asked

During the presentations and the evening cocktail, we heard many interesting ideas and took note of many appealling requests from the users:

– Light weight clients (John Michopoulos)
– Workflow and Files transferts (Eugenio Zimeo)
– Hierarchical Groups (Eugenio Zimeo)
– Kerberos (John Michopoulos)
– Mobility: move away as soon as possible (John Michopoulos)
– Wrapping: reuse and integrate previous work on Wrapping legacy code (Vladimir Getov)
– Virtual Nodes: make sure all nodes in a VN are running (Marco Aldinucci)
– Information on topology of nodes and virtual nodes (Eugenio Zimeo)
– You cannot force site to change and must adapt (Robert Jones)

3. The Grid

To be able to run experiments on Grid computing during our three day event, a Grid was built up, with the help of our different partners. The Grid was deployed on 20 different sites, in 12 different countries. We gathered a total of 473 machines, bearing 800 processors, totalling 100 GigaFlops (measured with the SciMark 2.0 agent for computing). This measure was taken using a pure JAVA benchmark. The advantage of JAVA is it's portability and ease of expression. On the other side, using native code runs faster, but the portability is lost, and more work has to be commited to deployment on heterogeneous platforms.

Difficulties encountered for the Grid setup

Since each site had it's own local configuration, nothing proved simple in the Grid configuration, as OS, JVM, schedulers, and protocols differed from one site to the other. But this was the work of people of the OASIS team, in particular Romain Quilici and Arnaud Contes, who prepared this Grid for the contest and the Plugtests. After a lot of hard work to set all the configuration files and accounts right, the deployment was made very simple and transparent to the Plugtests application developers, who had all the details hidden by the ProActive layer.

Installation:

We had to install ProActive on all sites. Some of these were using file systems (NFS) that allowed to install only once the software, and on others the instalation procedure had to be repeated on every machine. Since JAVA is the underlying programming language, it also had to be installed when it was missing, which meant finding the correct JAVA version depending on the operating system. Job schedulers used on different sites and that were not yet supported by ProActive were also implemented : PBS, SGE, OAR. Finally, acces configuration for all teams had to be performed on each site, which was found to be a very long and arduous task, due to the Grid's large size.

Security policy:

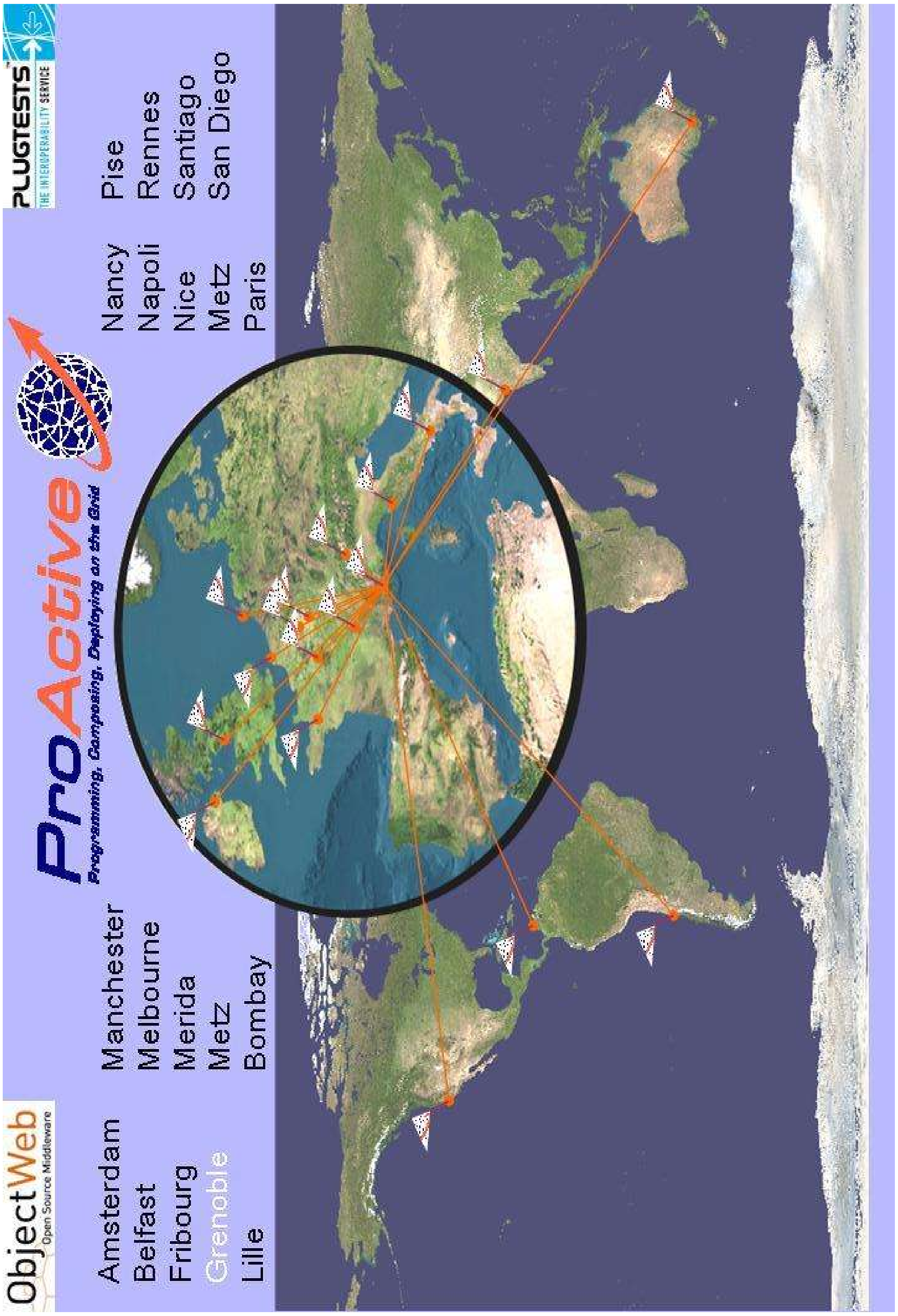
Each site had its own security policy, in which we define four levels of friendliness

- friendly : sites allowed all incoming/outgoing connections from/to machines on the ETSI Plugtests network.
- semi-friendly : sites allowed a range of ports to be open, in which case http was used as a communication protocol to gain acces to the machines.
- Semi-restrictive : sites allowed only ssh communication, and we had to implement and use ssh/RMI-tunneling to deploy jobs.
- Resctrictive : sites had a public IP address for the front-end, and private Ips for the backend nodes. Users were constrained to implement hierarchical deployment in their application. These sites were not very much used.

Some other problems were encountered, and forced other features to be added to ProActive. For example, the **proactive.useIPaddress** property was implemented to cope with sites that did not host a DNS, and **proactive.hostname** which dealt with machines that had two network interfaces.

The map which follows shows the location of the different sites. It shows how we reached a worldwide dissemination, with sites in Australia, Europe, North and South America, and India. A table is added to give technical details of every site, and sketch the computing power available and the differences of specifications.

ProActive Plugtests Grid : worldwide location of sites



ProActive Plugtests Grid : description of sites

Location	Machines	CPU (Ghz)	Processors	OS	Scheduler	JVM
Standard deployment						
Imag	70	0.9	2	Linux	OAR	Sun
Irisa	32	?	2	Apple	Web	Sun
Irisa	64	2.4	2	Linux	Web	Sun
LRI	50	2.2	1	Linux	ssh	Sun
Inria	16	2	2	Linux	LSF	Sun
Inria	19	0.93	2	Linux	LSF	Sun
San Diego	32	1.6	2	Linux x86_64	ssh	1.5
ESSI	17	2.26	1	Linux	ssh	Sun
ESSI	12	2.8	1	Linux	ssh	Sun
Man-chester	20	3	2	Linux	Globus	Sun
Amster-dam	20	1 (P3)	2	Linux	prun	Sun
Chile	11	1.5(P4)	1	Linux	ssh	Sun
Belfast	2	1.5	1	Linux	ssh	Sun
Supelec	8	2.4	1	Linux	ssh	Sun
LIFL	4	1.7|1.9|3|3	1	Linux	ssh	Sun
ISTI	4	1.8|2|2|2	1	Linux	ssh	Sun
Bull	2	1.3	16|16	Linux ia 64	ssh	Bea
EIF	25	0.5	1	Solaris9	ssh	Sun
Loria	6	1.5	1	Linux	ssh	Sun
Loria	2	0.7	32|24	SGIIrix	ssh	Sgi
Napoli	5	2.4	2	Windows	ssh	Sun
Hierarchical deployment						
Mel-bourne	13	2.4	1	Linux	PBS	Sun
Napoli	7	2.2	1	Linux	SGE	Sun
Pise	20	0.8	1	Linux	no	Sun

The table above lists the technical details of the different sites. The hierarchical deployment had to be used when the backend machines were not directly accessible. Full deployment was achieved by using first an xml file to deploy on the frontend, which in turn reads another xml file to create JVMs on backend machines.

4. The Contest

The second and third days hosted two different activities. A contest, involving finding solutions to the N-Queens problem was organised, and Hands-on tutorials were also provided.

The Hands-on session was dedicated to testing the deployment of ProActive applications on various Grids, using various standard protocols (ssh, Globus, Web Services, Jini, LSF, PBS, rsh, rlogin, etc.). Tutorials were also on the schedule, and some members of the ProActive development team offered their help to newcomers with the guided tour. Theses activities were set in the same time span as the contest, which was meant to get people thinking on the same problem, with the cleverest implementation rewarded.

On Wednesday morning, a talk, titled “NTU Research Activities and N-Queens findings", was given Yuh-Pyng (Arping) Shieh, from NTU National Taiwan University.

4.1 The N-Queen Challenge

A contest was organised around solving an embarassingly parallel problem. Using ProActive, for the largest chessboard of dimension N, count the number of solutions for placing N non-threatening queens. The world record is for N=24, having 227,514,171,973,736 solutions.
All users were asked to use ProActive as their middleware, and could freely use the power of the 800+ processors which were dispatched around the world.

4.2 The teams

The teams were composed of the following people:

AlgoBAR :
Sylvain BELLINO
Philippe HENRI
Philippe SIGAUD

DCC.UCHILE.CL : Dpto. Cs. de la Computación, Universidad de Chile
Antonio Cansado
Marco Danelutto
Mario Leyton
Luis Mateu

INRIA :
Vincent CAVE
Guillaume CHAZARAIN
Alexandre Di Constanzo
Bernard Serpette

NTU :
Arping, Dr. Yuh-Pyng Shieh

TOURNANT :
Christophe TOURNANT

University of Southern California :
Yu Chen
Laurent Devallonné
Sébastien Flament
Eugene Song

4.3 The rules

This event was strictly an engineering event, not a conference, nor a workshop. As such, an active participation was requested from the companies/organisations which had to write their own implementation of the N-Queens algorithm, and eventually modify existing xml deployment to adapt to their strategy.

There was no compulsory programming language, but all teams used JAVA to write their code, expect from the Arping team which hid some native routines inside a JAVA wrapper. Arping's scheme led to a faster algorithm, but lost JAVA's portability. In this last case, sites had to be updated with native code, which would be hard to do on a bigger scale. On the other hand, the all-JAVA approach allowed for transparent migration of code to distant nodes, with no manual code exportation.

The criterion for deciding the winners were based on
- the greatest number of solutions found
- the biggest number of processors used
- the fastest algorithm.

4.4 The timings

Every team was given enough time to complete several runs of their algorithm. In the following table, the results are shown in a synthetic way : for every N queens computation, the number of solutions found by the team is displayed.

ARPING		TOURNANT		AlgoBAR	
18	666090624	20	39029188884	21	314666222712
18	666090624	<21	3933929944	19	4968057848
18	666090624			19	4968057848
19	4968057848				
20	39029188884				
Total	45 995 518 604		42 963 118 828		324 602 338 408
UCHILE		USC		INRIA	
		15	2279184	21	39029188884
18	**666090624**	18	666090624	20	314666222712
4 x 20	**156116755536**	19	4968057848		
2 x 21	**629332445424**				
2 x 19	**9936115696**				

Total	**796 051 407 280**		5 636 427 656		353 695 411 596

4.5 The results

The Chilean team got ahead of the other 5 participants. They found the number of solutions for 18 queens, 19 queens twice, 20 queens 4 times and 21 queens once, in an hour. They were the best when considering

- the number of solutions found in one hour (800 billion),
- the number of nodes used (560),
- and the speed of the algorithm (21 queens in 24'38”).

The winners, Mario Leyton, Antonio Cansado, and Luis Mateu from Chile.

5. Conclusion and Future Events

The Plugtests, coorganised by INRIA and ETSI, pleased all the participants. It was an event both useful for the users, who received help from the ProActive developers, and the OASIS team, who received feedback from the users. We were forced to add functionnalities to the middleware to be ready and effective for the Plugtests, and we now have a complete and stable system. We had to develop certain aspects that had been left out, due to time restrictions and priorities, that were in fact of primary importance. We are also very satisfied with the results obtained during the N-queens contest, which showed that applications could take advantage of the Grid in a simple way. Another happy discovery was the number of different scientific domains which could use our middleware in their applications – this is a direct effect of the generic programming model used inside, which can be reused for biology, physics and evolving phenomenons.

We did have trouble setting up the Grid to work, as some protocols had to be implemented, so that the clusters could be reached, even with their restrictive security policies, different OSs and local configuration. This difficult tuning proves that the Grid is not yet ready for Plug & Play, but once this configuration was achieved, the work for the users was simple. Indeed, the deployment on the different sites was not a source of problems, which is an indicator of how ProActive is fit for usage, as users were not bothered by system configuration, and could instead focus on the internals of their application.

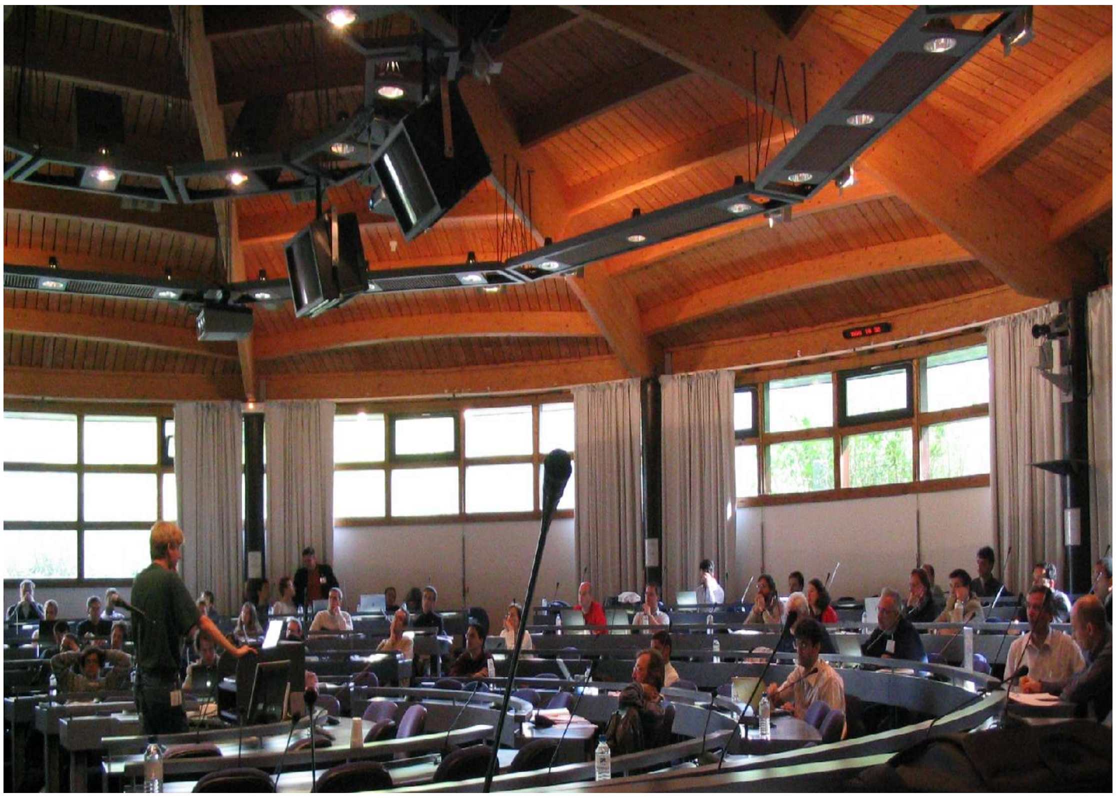
Pressed by the general demand, we will be organising another Plugtests on October 10-14th 2005. The event is planned to be larger, on all scales: we expect more people (more than 150), a longer time span (5 days), a larger Grid, the use of other middleware, and an even wider panel of domains. This future event will be coorganised with several European Projets. The application used for the contest and interoperability Plugtests is not yet fixed, but we have been thinking about a travel sales man problem, which needs many more communications, and might be even more interesting and demanding to supervise.

For any questions, please contact Isabelle Attali :
ia@sophia.inria.fr

Thanks to all of the participants of the 1st Grid Plugtests edition

Isabelle Attali
Denis Caromel
Thierry Priol
Yvon Jegou
Franck Cappello
Vincent Neri
Jean-Louis Roch
Eric Ragusi
Daniel Terrer
Francis Montagnac
Marc VESIN
Franck Yampolsky
David Geldreich
Sebastien GEORGET
Nicolas Niclausse
Regis Daubin
Antoine Zogia
Alain Giulieri
Jean-Louis Faraut
Christophe Coroyer
Michel Riveill
Le Rouzic
Stephane Martin
Tony Reix
Simon Derr
Jean-Claude Paul
Jens Gustedt
Xavier Cavin
Olivier Demengeon
Benjamin DEXHEIMER
Stéphane Vialle
Arnaud Contes
Patrick Mercier
Melab Nouredine
El-Ghazali Talbi

Sebastien CAHON
Henri Bal
Kees Verstoep
Stephen Pickles
Matt Ford
Mike Jones
Ron Perrott
Tony McHale
Philip Preston
Pierre Kuonen
Jean-François Roche
Ranieri Baraglia
Giancarlo Bartoli
Domenico Laforenza
Marco Danelutto
Pietro Vitale
Marco Vanneschi
Marco Aldinucci
Eugenio Zimeo
Nadia Ranaldo
Luis Mateu
Jose Piquer
Andrew A. Chien
Troy Chuang
Rajkumar Buyya
Chee Shin Yeo
Leandro León
Gilberto Diaz
José Aguilar
Romain Quilici
Laurent Devallonné
Maria-Luisa Parlatano
Patrick Guillemin
Philippe Cousin
The OASIS team


The theater

ProActive User Group participants

The demo

The panel of experts