Grid Component Model and ProActive Parallel Suite

Bridging Distributed and Multi-Core Computing

Denis Caromel, et al.

http://ProActive.ObjectWeb.org

OASIS Team

INRIA -- CNRS - I3S -- Univ. of Nice Sophia-Antipolis, IUF

October 14th, CGW, Kraków, Poland

1. **Background: INRIA, OASIS, ActiveEon**
2. **ProActive Parallel Suite & Active Objects**
3. **Components and Standardization (GCM)**
4. **Applications and Perspectives: SOA+GRID**

1. Background

INRIA and OASIS Team

Computer Science and Control

- 8 Centers all over France
- Workforce: 3 800
- 186 Million Euro annual budget
- Strong in standardization committees:
  - IETF, W3C, ETSI, …
- Strong Industrial Partnerships

- Foster company foundation:

  90 startups so far

  - Ilog (Nasdaq, Euronext)
  - …
  - ActiveEon

A joint team between:

- INRIA, University of Nice- CNRS

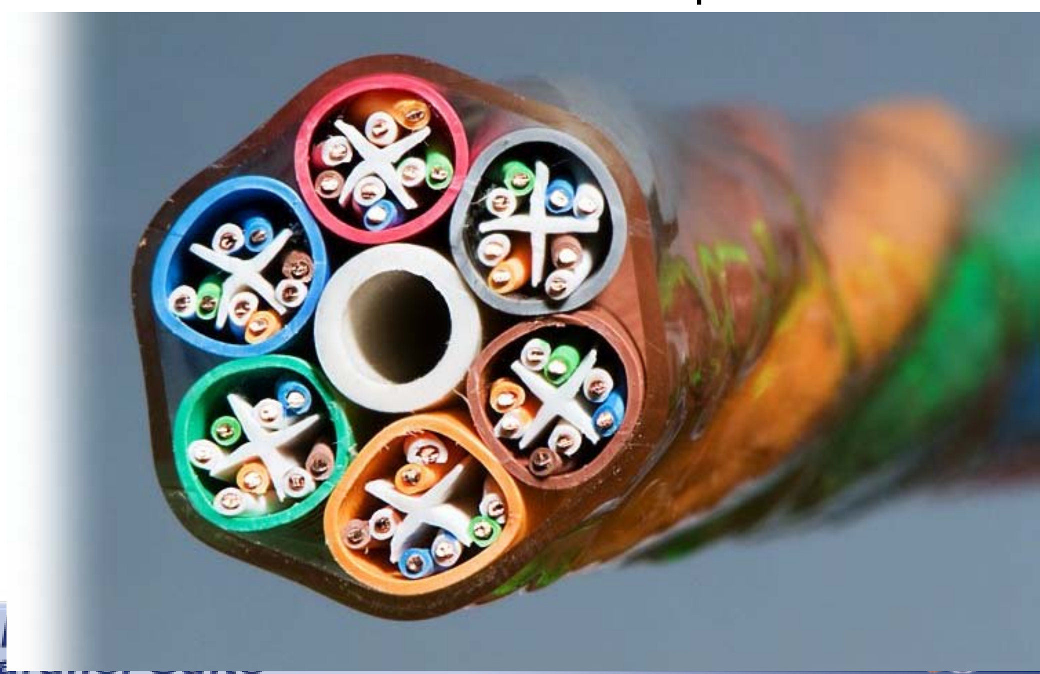
•Created in 1999

•Started the *ProActive Parallel Suite*

•Over 40 persons

•Distributed and Parallel:

From Multi-cores to Enterprise GRIDs



ACTIVEEON
SCALE BEYOND LIMITS

Startup Company Born of INRIA

Co-developing, Providing support for Open Source ProActive Parallel Suite

Winner 80/1000 applications (Minister of Research Contest)

Several Customers (Worldwide: Boston USA, etc.)

2. ProActive Parallel Suite

&

Active Objects

ProActive Parallel Suite

- Written in Java
- Features:

1. Eclipse GUI
2. Parallel+Dist. Progr.
3. Scheduling & Grids

Used in production by industry

ProActive Parallel Suite

PROGRAMMING

Java Parallel Frameworks
for HPC, Multi-Cores, Distribution, Enterprise Grids and Clouds.

Featuring: Async. comms, Master-Worker, Monte-Carlo, SPMD, components and legacy code wrapping.

OPTIMIZING

Eclipse GUI (IC2D)
for Developing, Debugging, Optimizing your parallel applications.

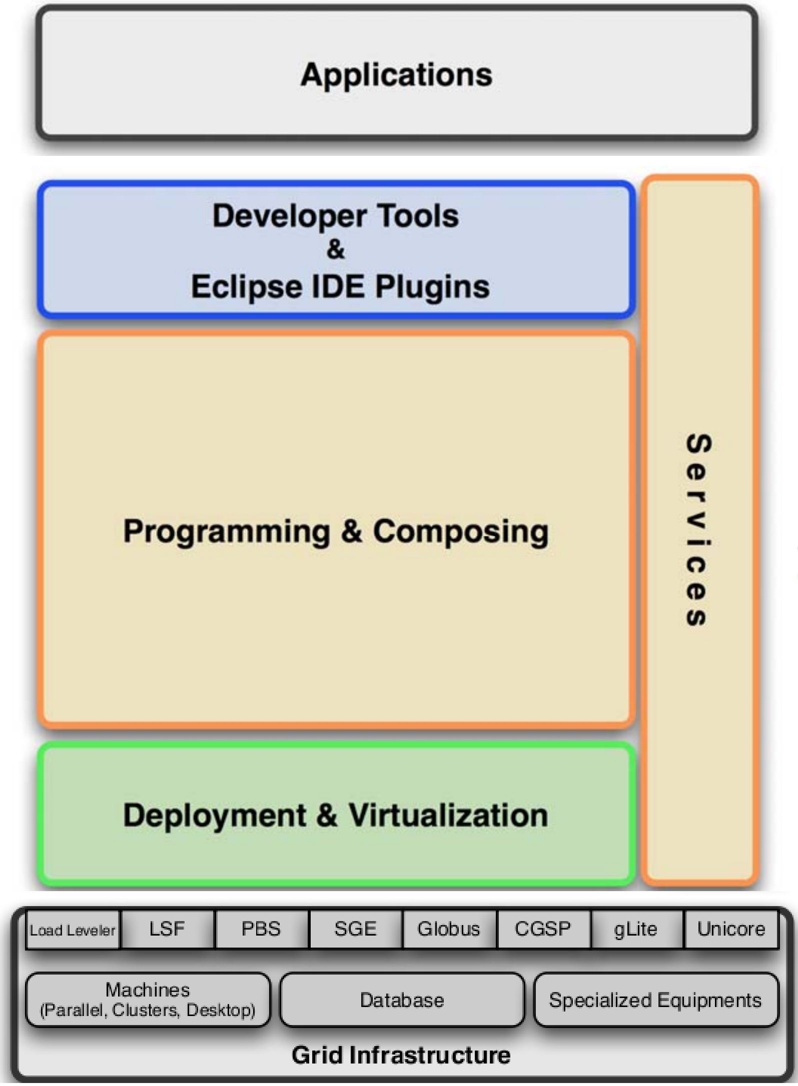
Featuring: graphical monitoring and benchmarking with report generation.

SCHEDULING

Multi-Language Scheduler
for Workflows made of C, C++, Java, Scripts, Matlab, Scilab tasks.

Featuring: graphical user interface, resource acquisition and virtualization.

ProActive Parallel Suite (1)



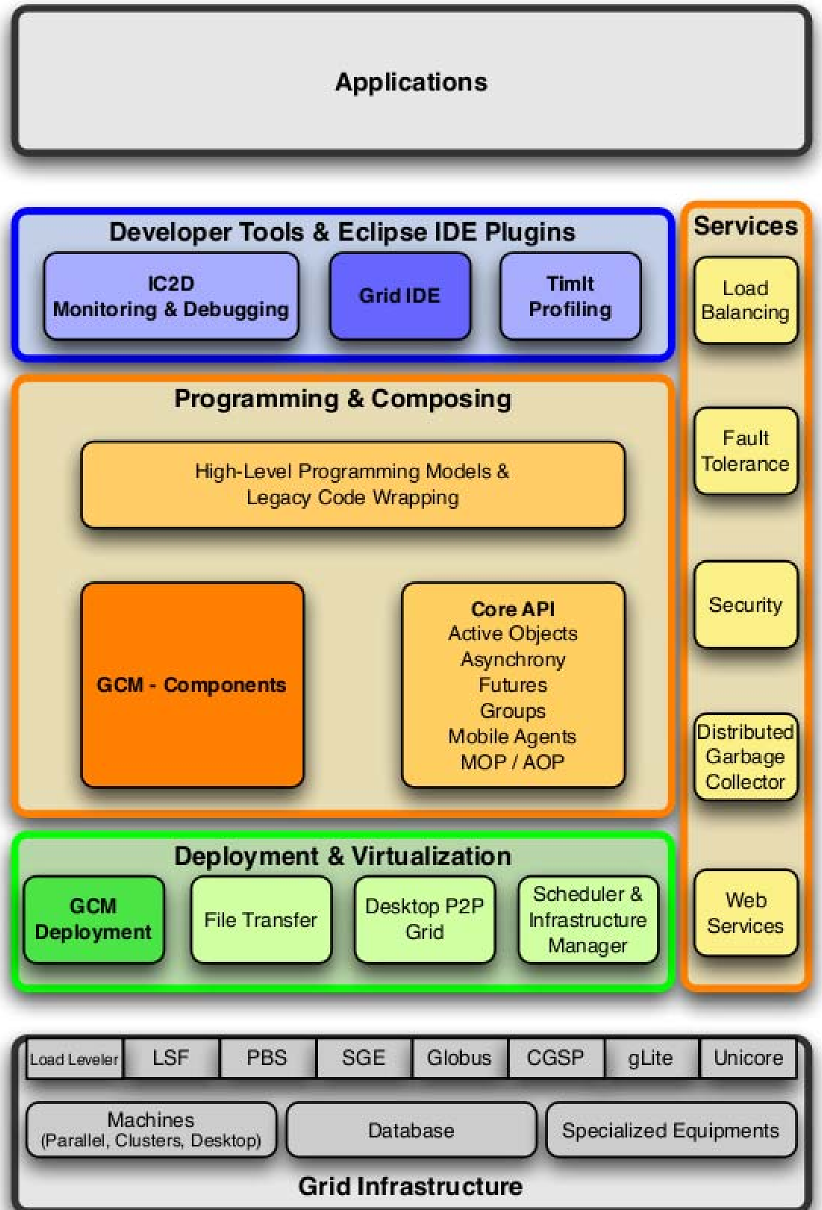
ProActive
Programming, Composing, Deploying on the Grid

Open Source

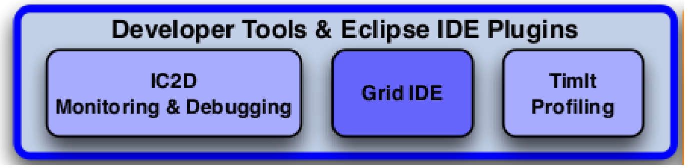
+

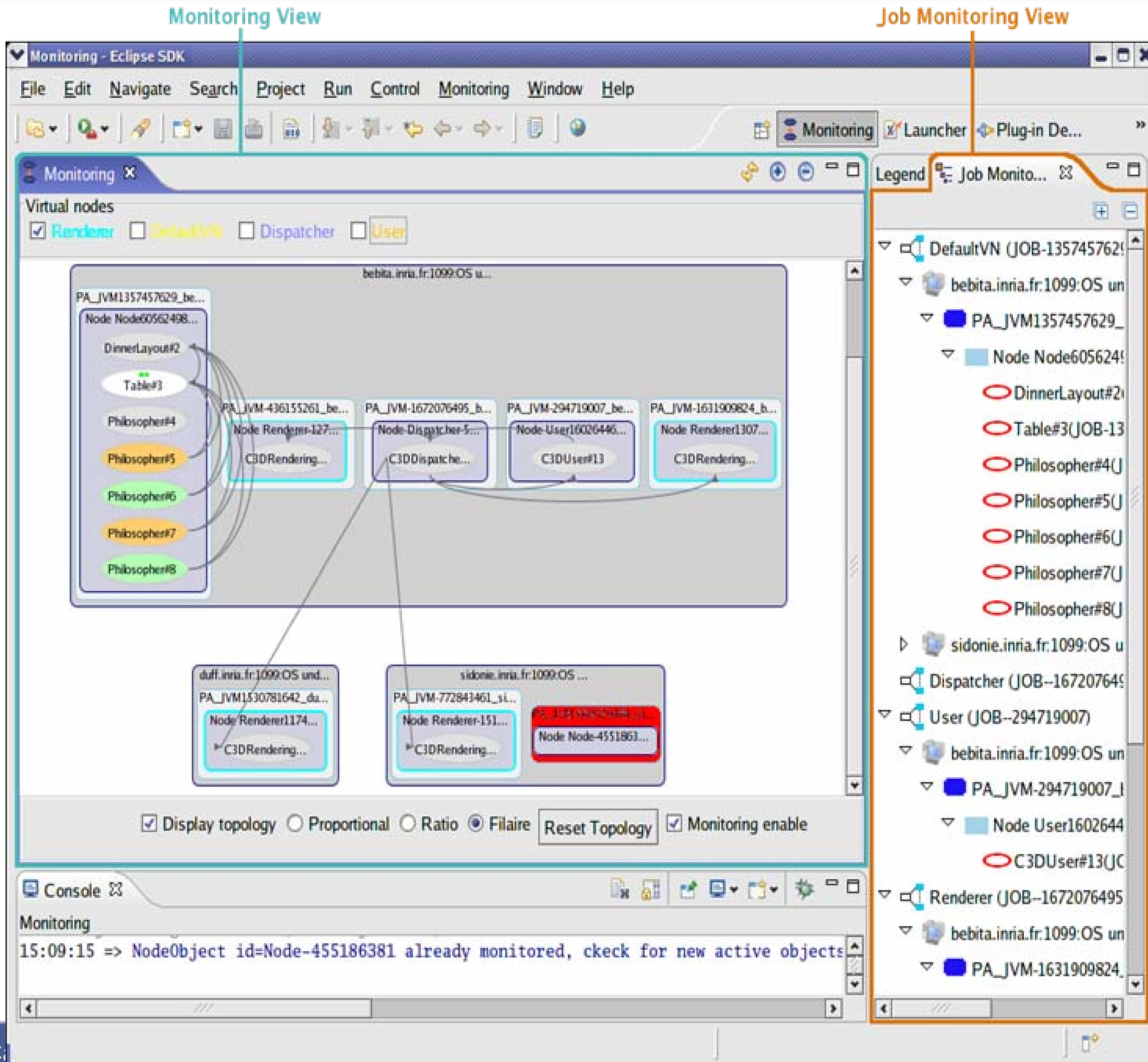
PROFESSIONAL SUPPORT

ProActive Parallel Suite: GUI

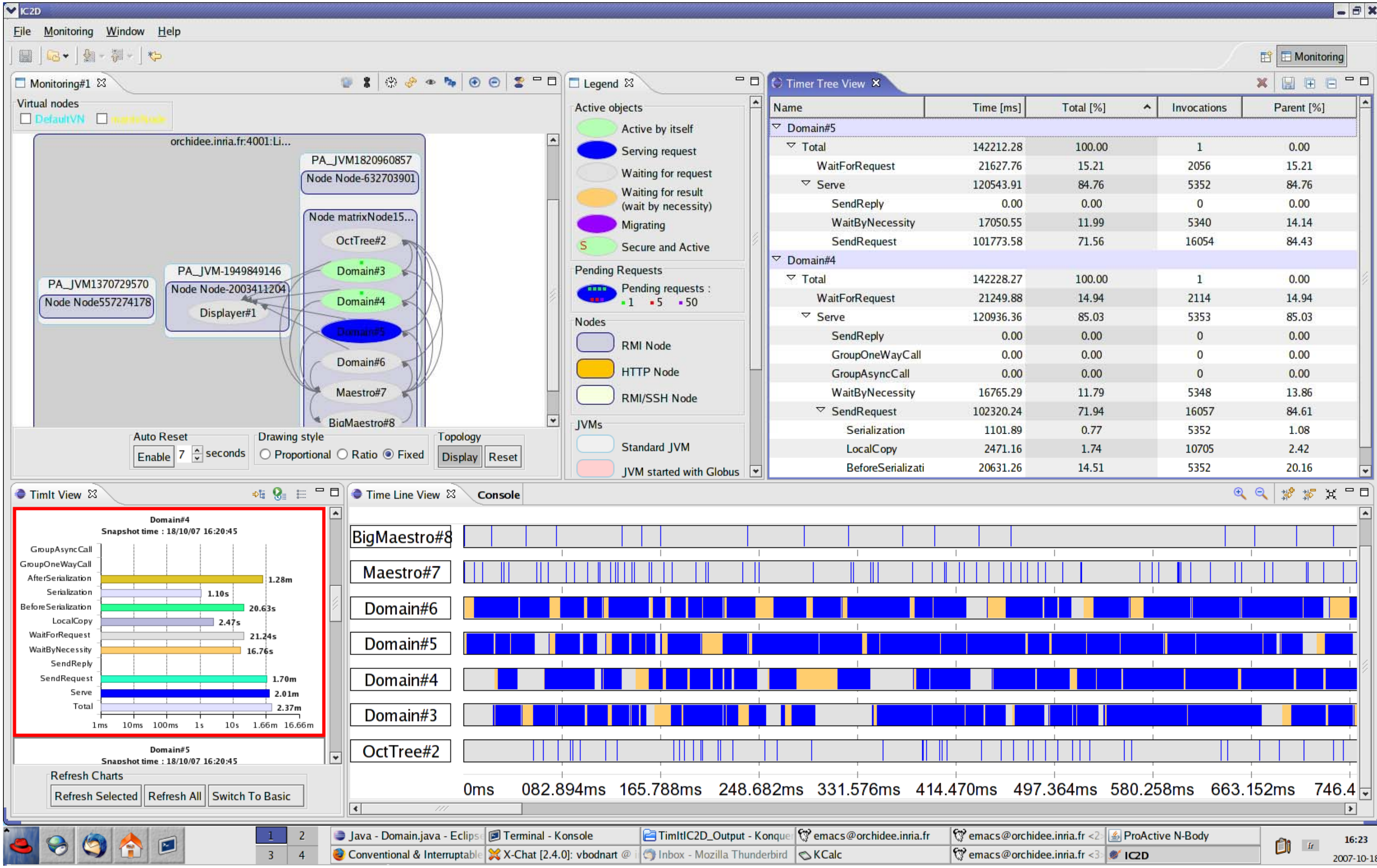


ProActive Parallel Suite: GUI




Monitoring View
Job Monitoring View
Monitoring - Eclipse SDK
File Edit Navigate Search Project Run Control Monitoring Window Help
Monitoring
Launcher
Plug-in De...
Monitoring
Virtual nodes
Renderer
DefaultVN
Dispatcher
User
Display topology
Proportional
Ratio
Filaire
Reset Topology
Monitoring enable
Console
Monitoring
15:09:15 => NodeObject id=Node-455186381 already monitored, ckeck for new active objects
Legend
Job Monito...
DefaultVN (JOB-135745762
bebita.inria.fr:1099:OS un
PA_JVM1357457629_
DinnerLayout#2
Table#3(JOB-13
Philosopher#4(J
Philosopher#5(J
Philosopher#6(J
Philosopher#7(J
Philosopher#8(J
sidonie.inria.fr:1099:OS u
Dispatcher (JOB--16720764
User (JOB--294719007)
PA_JVM-294719007_
Node User1602644
C3DUser#13(JC
Renderer (JOB--1672076495
PA_JVM-1631909824_

IC2D



ProActive Parallel Suite

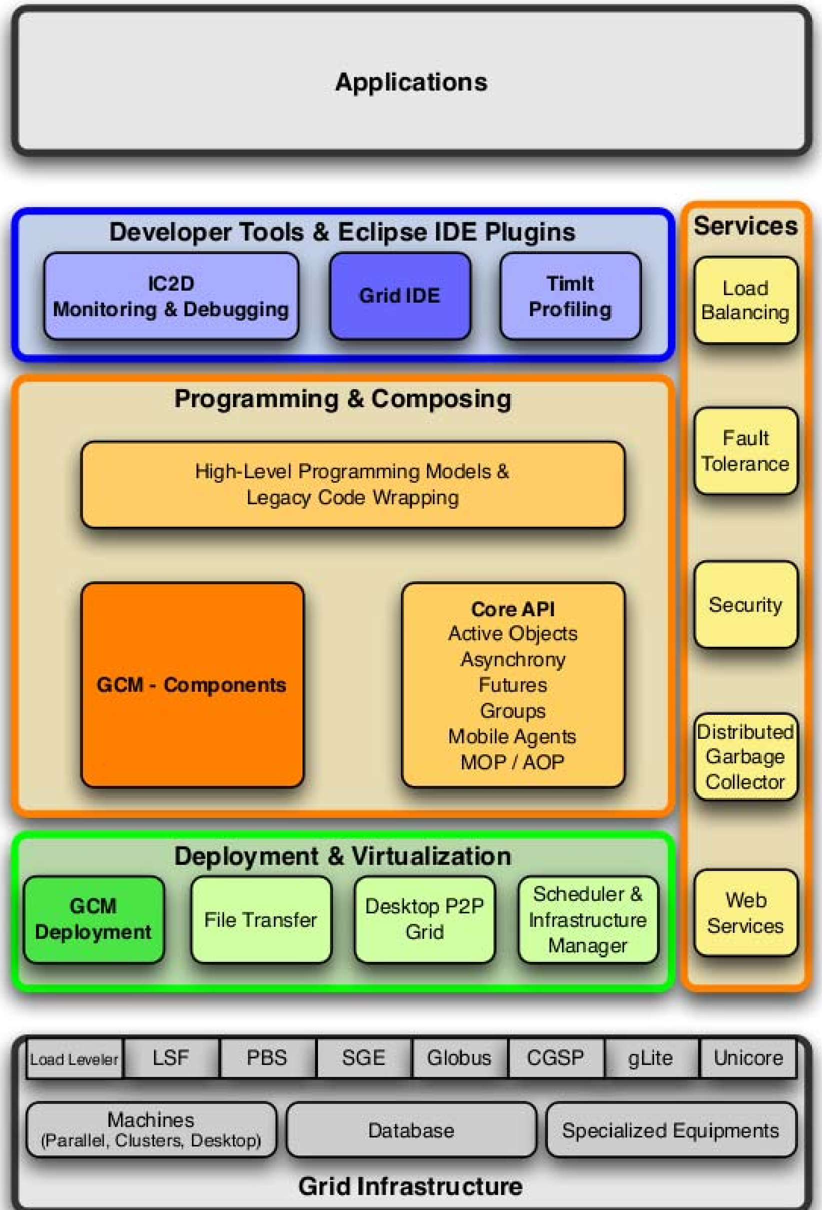
ChartIt


PA_JVM251111462
Charts
Chart#1 [LoadedClassCount]
2,5 k
2,4 k
2,3 k
2,2 k
2,1 k
2,0 k
LoadedClassCount
Chart#3 [ThreadCount]
40
35
30
25
ThreadCount
Chart#2 [UsedHeapMemory]
7,0 M
6,0 M
5,0 M
4,0 M
UsedHeapMemory
Chart#4 [InternalBodiesCount, UserBodiesCount]
5
4
3
2
1
0
InternalBodiesCount
UserBodiesCount
Chart#5 [ThreadCount] Meter
0
5
10
15
20
25
30
35
ThreadCount
35.0
Chart#6 [AllBodiesCount, InternalBodiesCount, UserBodiesCount, HalfBodiesCount]
7
6
5
4
3
2
1
0
AllBodiesCount
InternalBodiesCount
UserBodiesCount
HalfBodiesCount

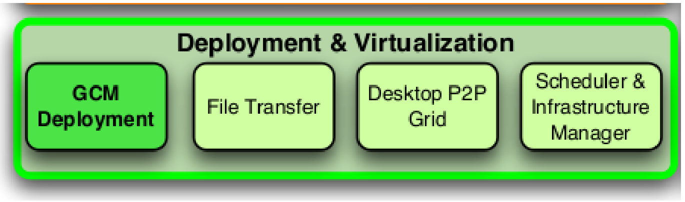
Video 1: IC2D

Monitoring, Debugging, Optimizing

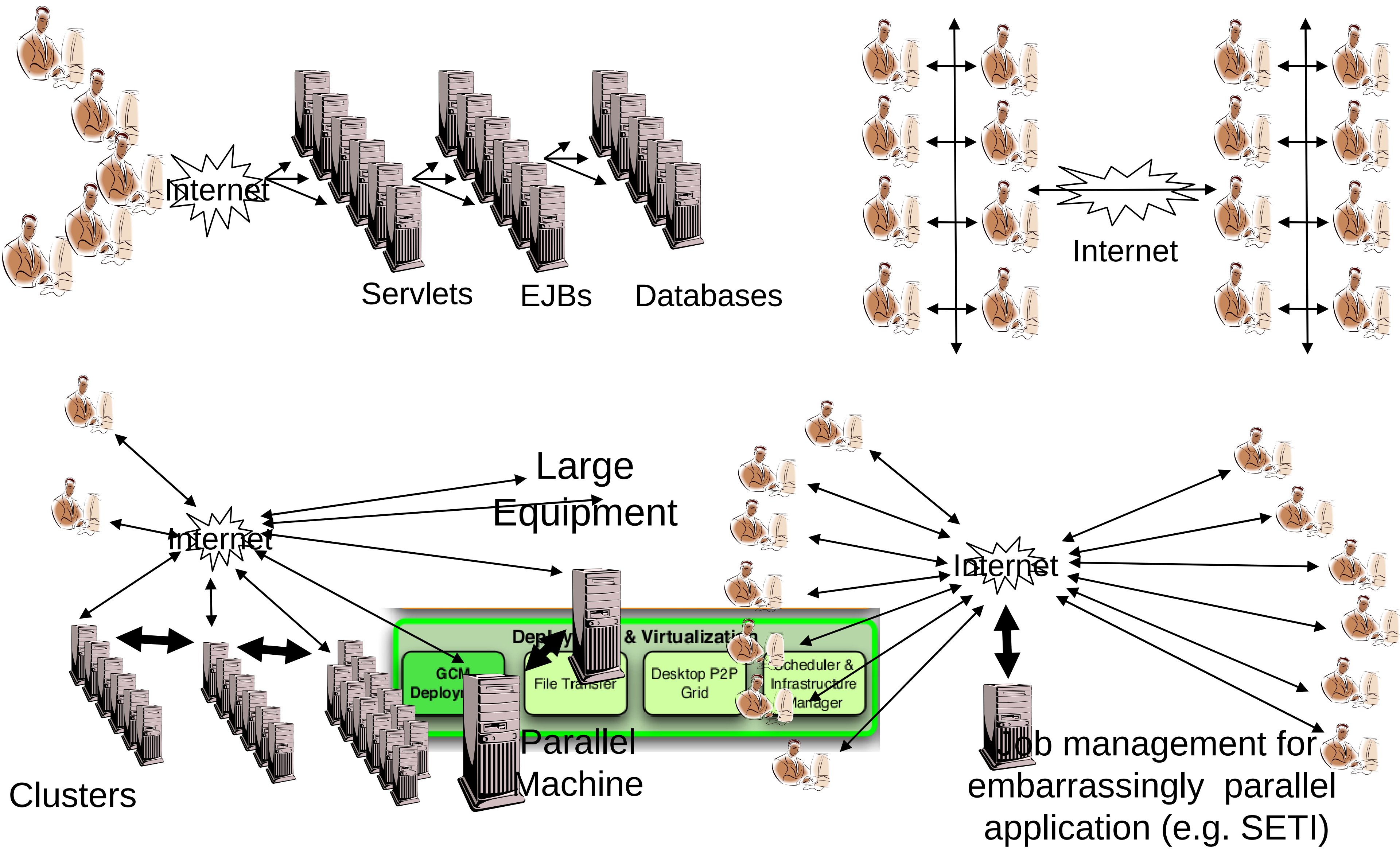
ProActive Parallel Suite: Deploy



ProActive Parallel Suite: Deploy

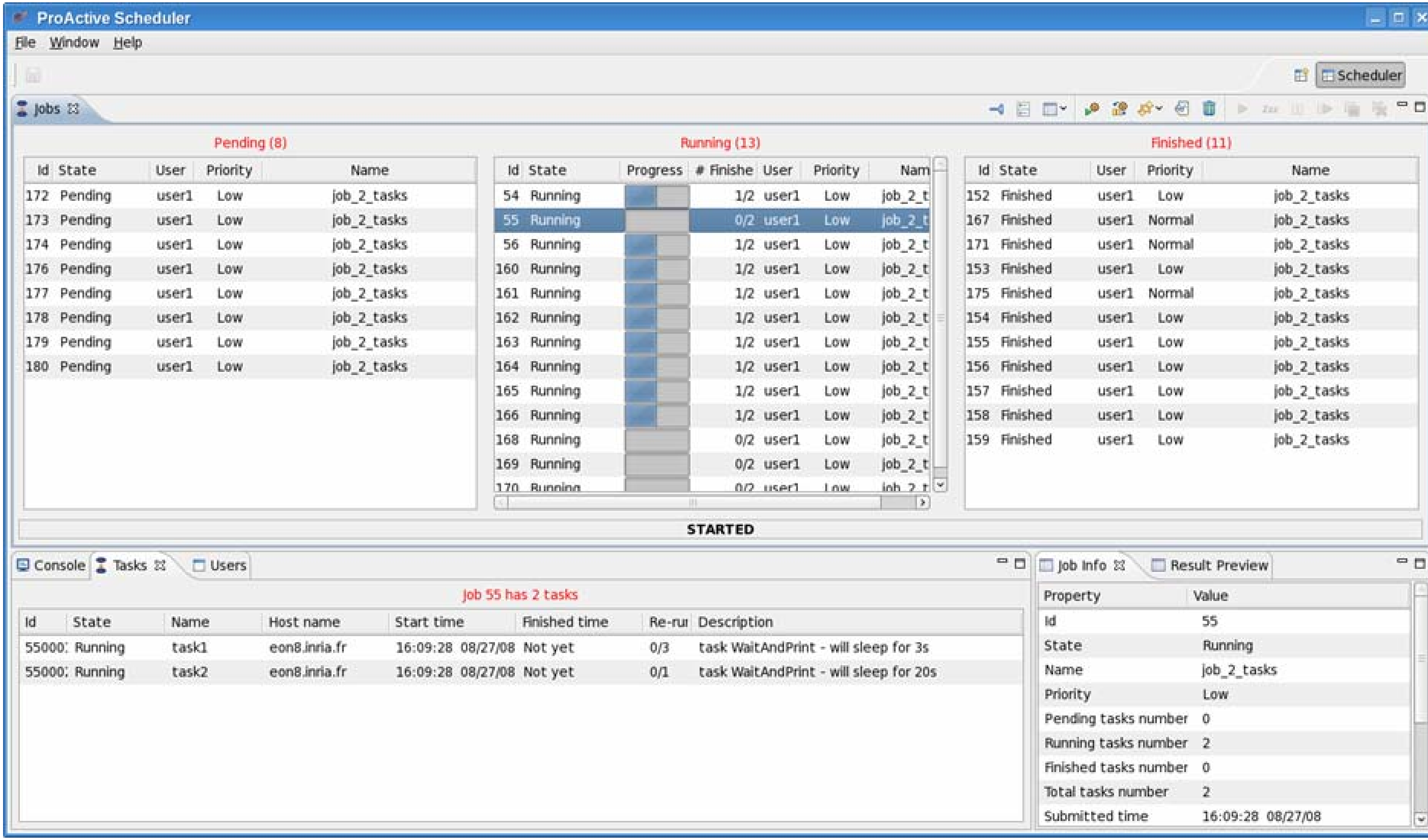


Deploy on Various Kinds of Infrastructures



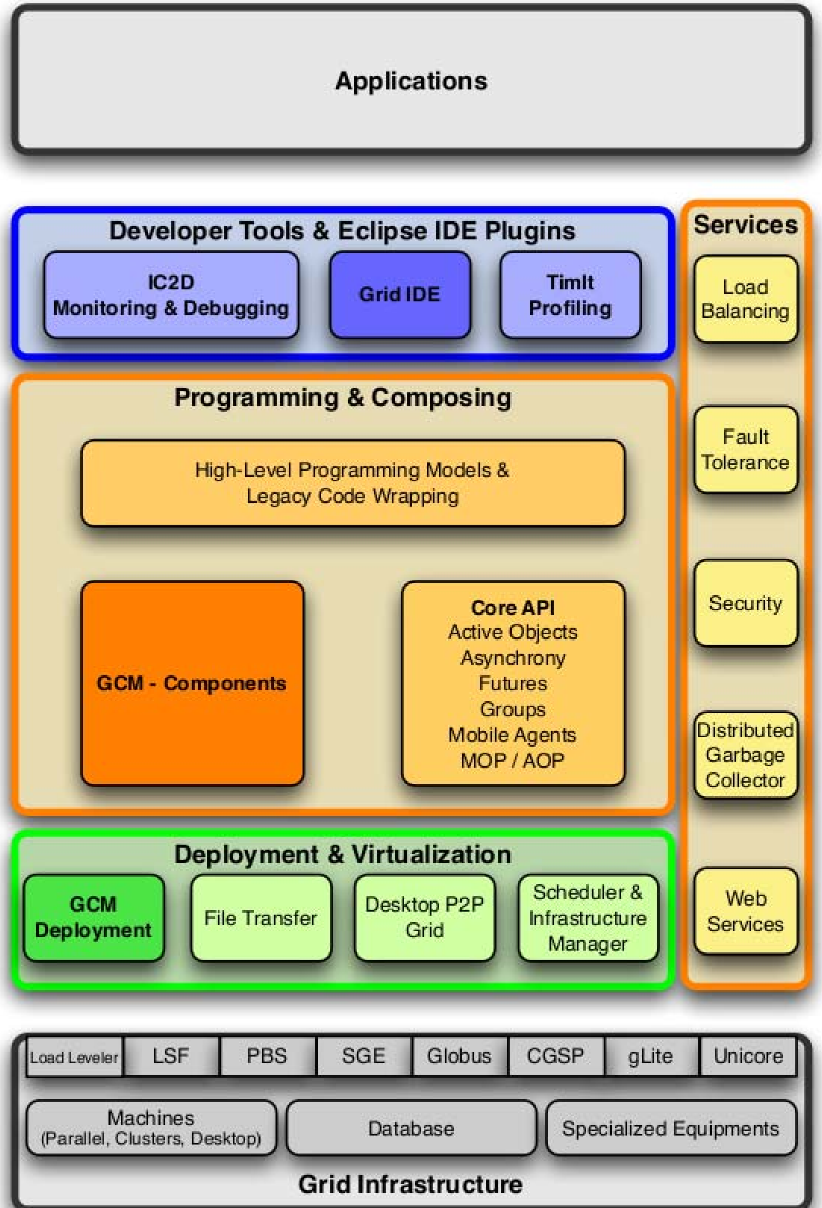
Scheduler and Resource Manager: User Interface

Scheduler: User Interface

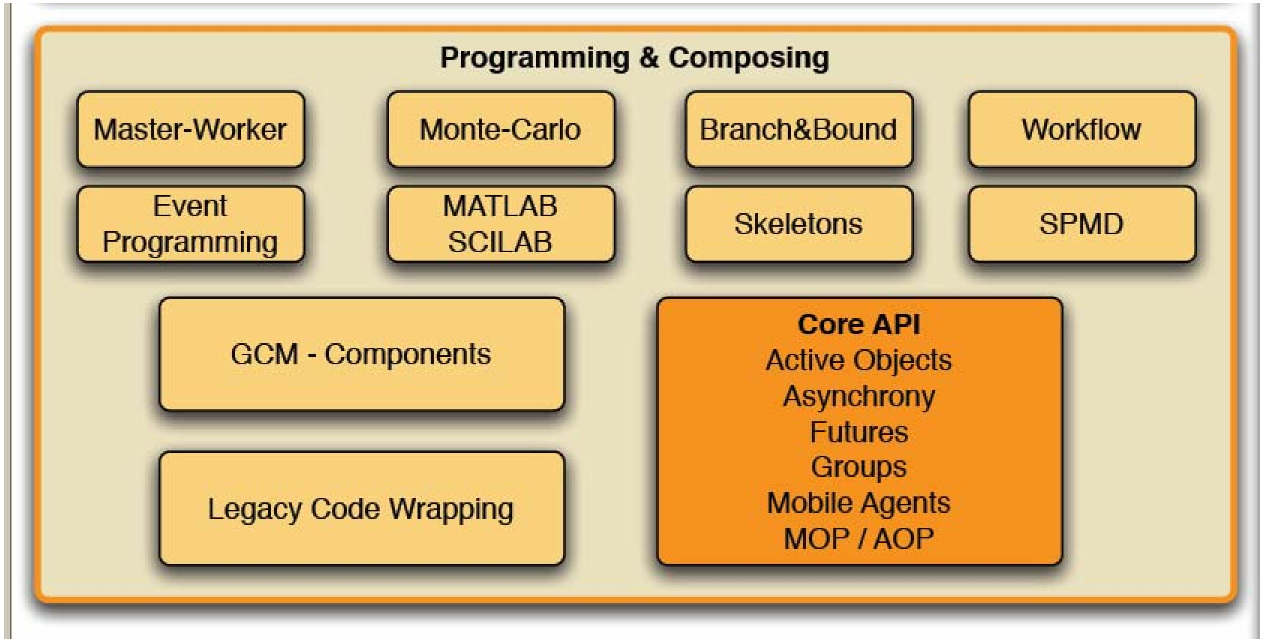


Video 2:
Scheduler, Resource Manager

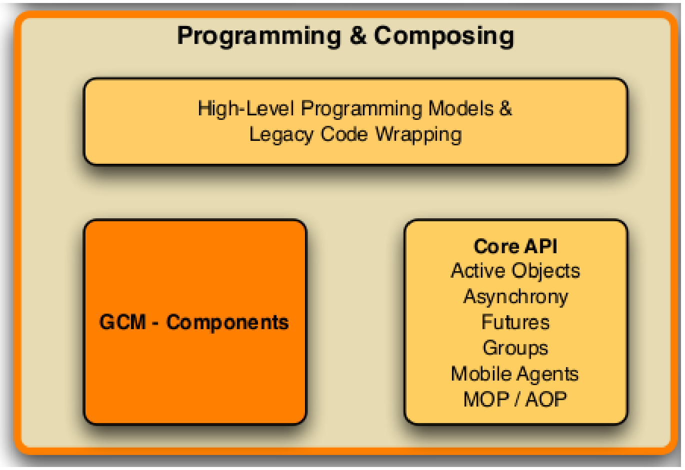
ProActive Parallel Suite: Program



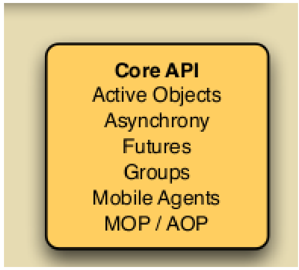
ProActive Parallel Suite



ProActive Parallel Suite: Program



ProActive Parallel Suite: Program

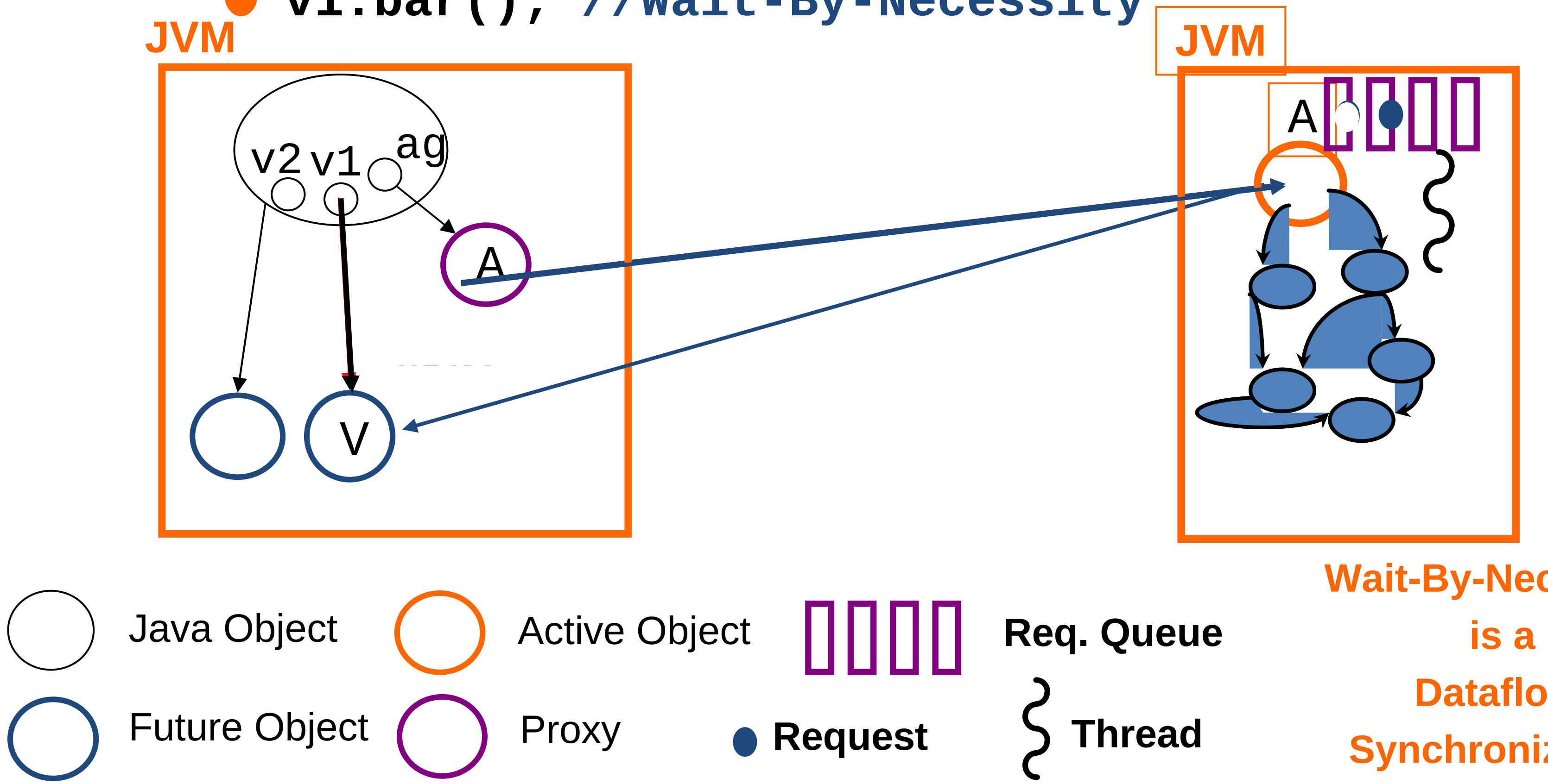


Distributed and Parallel

Active Objects

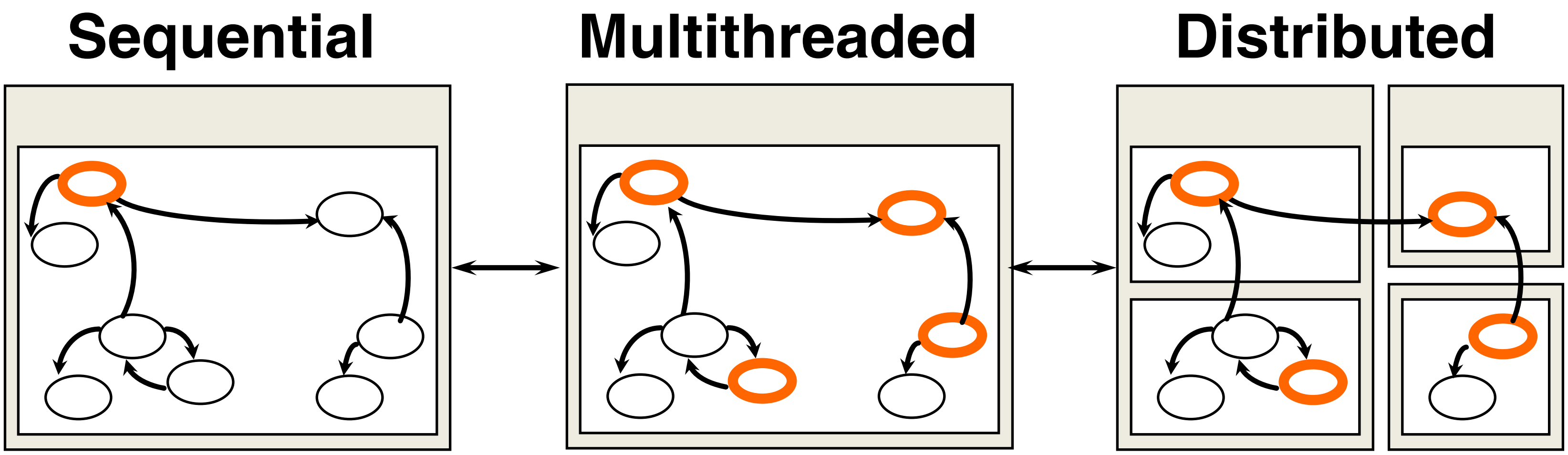
ProActive : Active objects

```
A ag = newActive ("A", […], VirtualNode)
V v1 = ag.foo (param);
V v2 = ag.bar (param);
...
v1.bar(); //Wait-By-Necessity
```



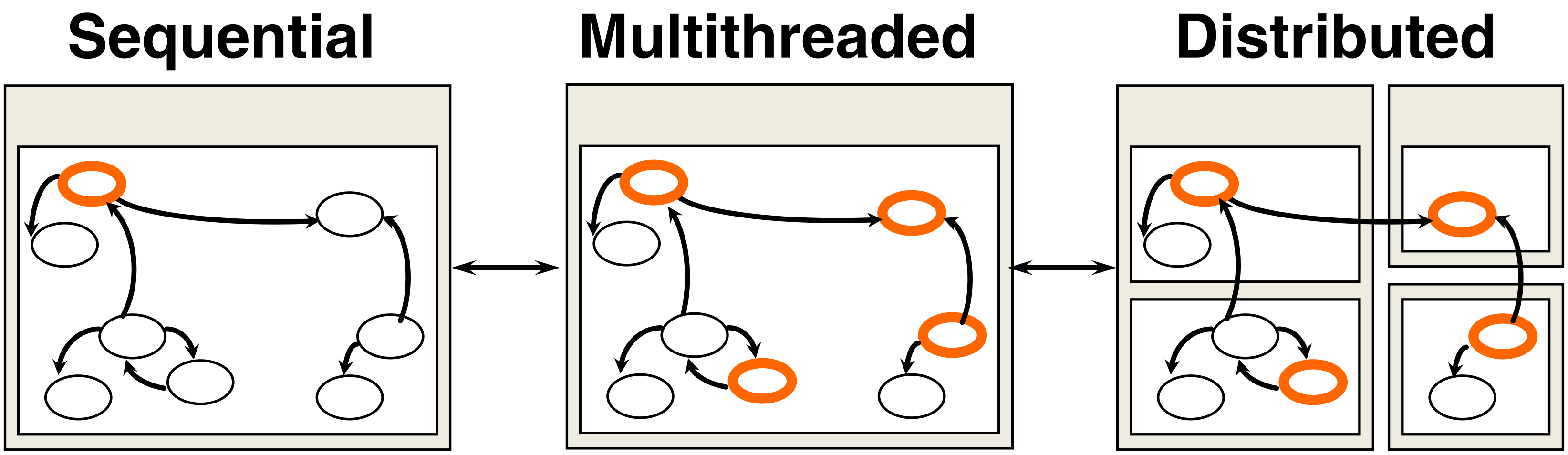
Wait-By-Necessity is a Dataflow Synchronization

ProActive:
Inter- to Intra- Synchronization



Synchronizations, Behavior: not dependent upon the physical location (mapping of activities)

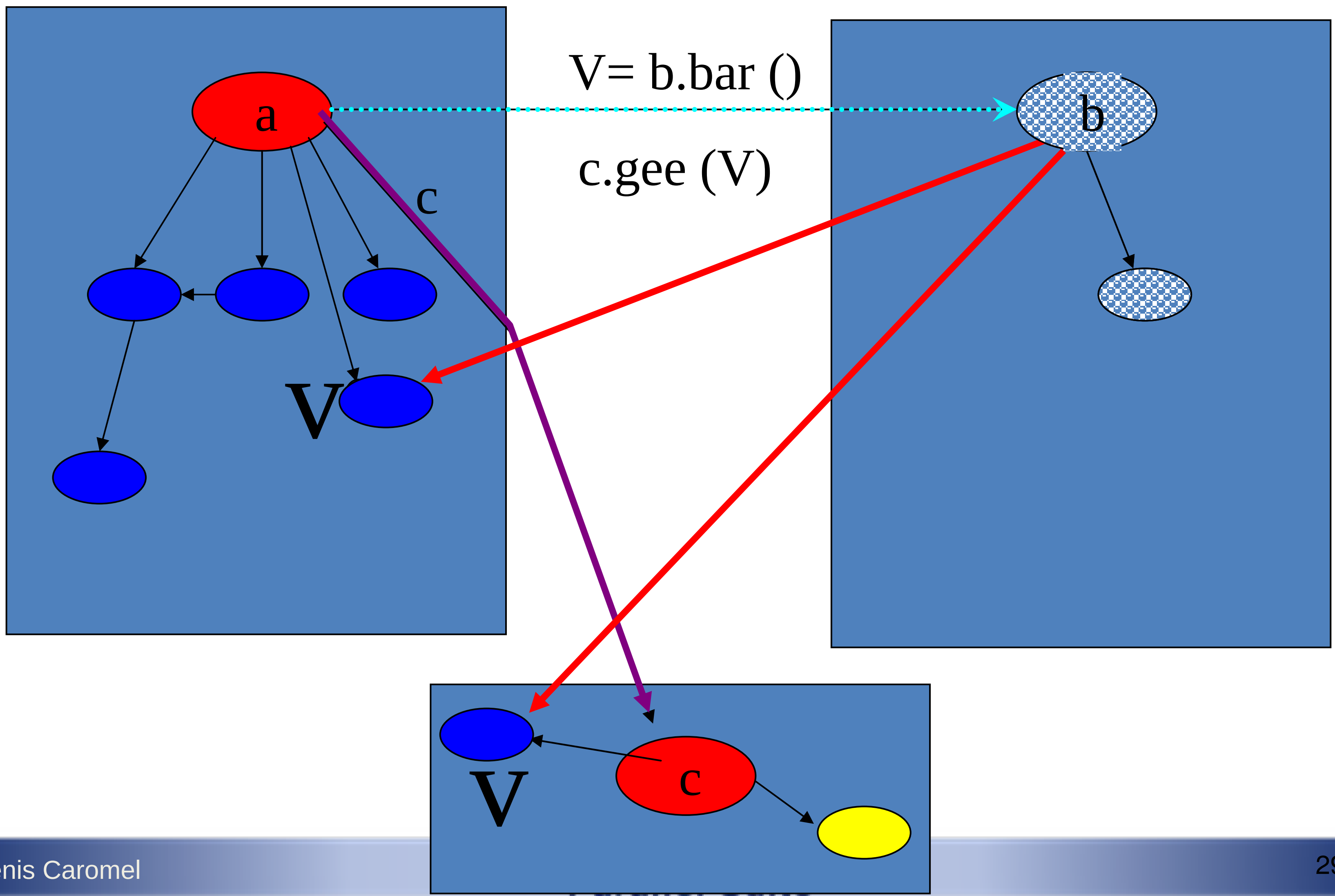
ProActive:
First-Class Futures



Synchronizations, Behavior: not dependent upon the physical location (mapping of activities)

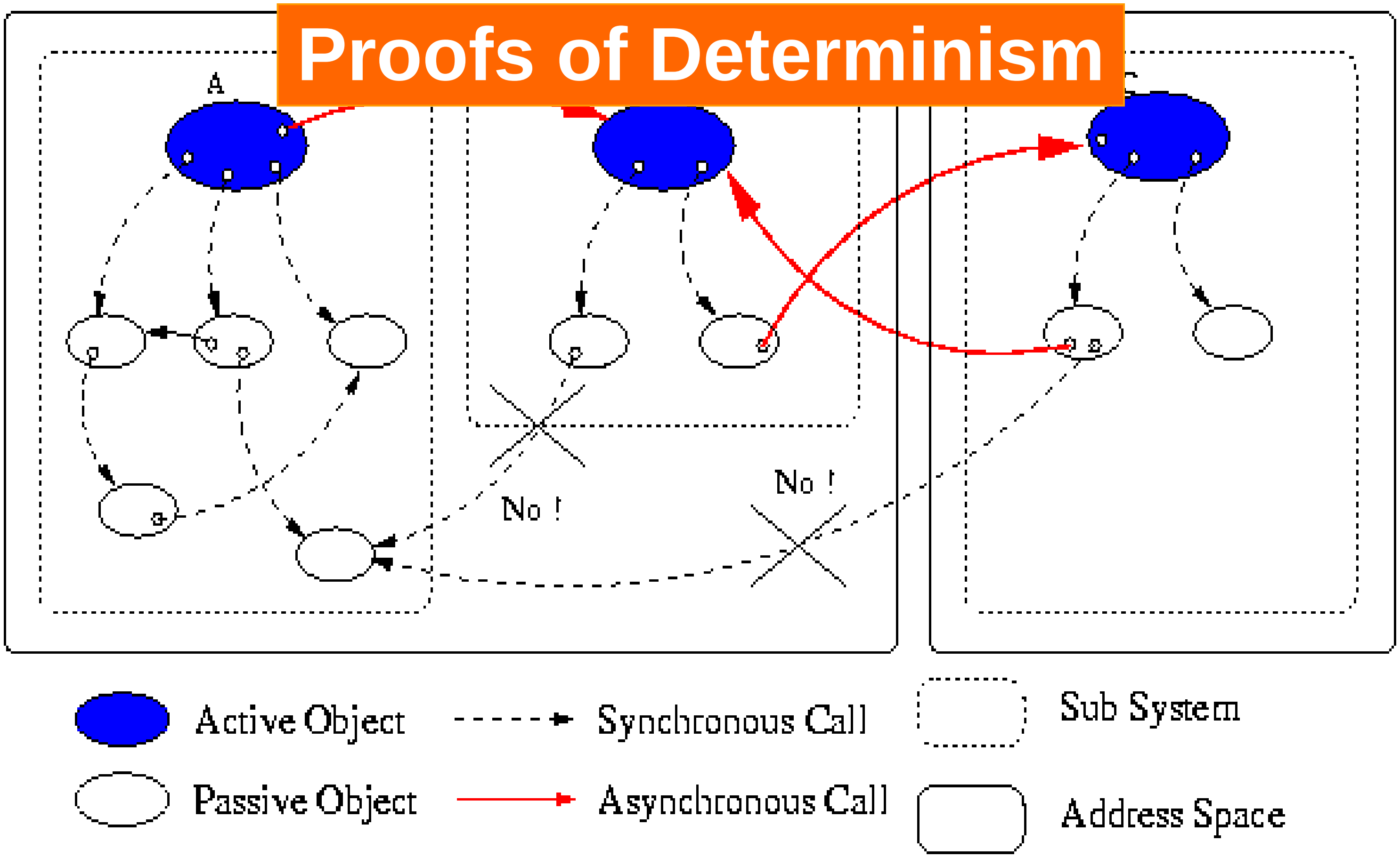
Wait-By-Necessity: First Class Futures

Futures are Global Single-Assignment Variables



Standard system at Runtime: No Sharing

NoC: Network On Chip



Calculus

ASP:
Asynchronous Sequential Processes

Proofs in GREEK

$$\frac{(a,\sigma)\to_S(a',\sigma')}{\alpha[a;\sigma;\iota;F;R;f]\parallel P\longrightarrow\alpha[a';\sigma';\iota;F;R;f]\parallel P}\ (\text{LOCAL})$$

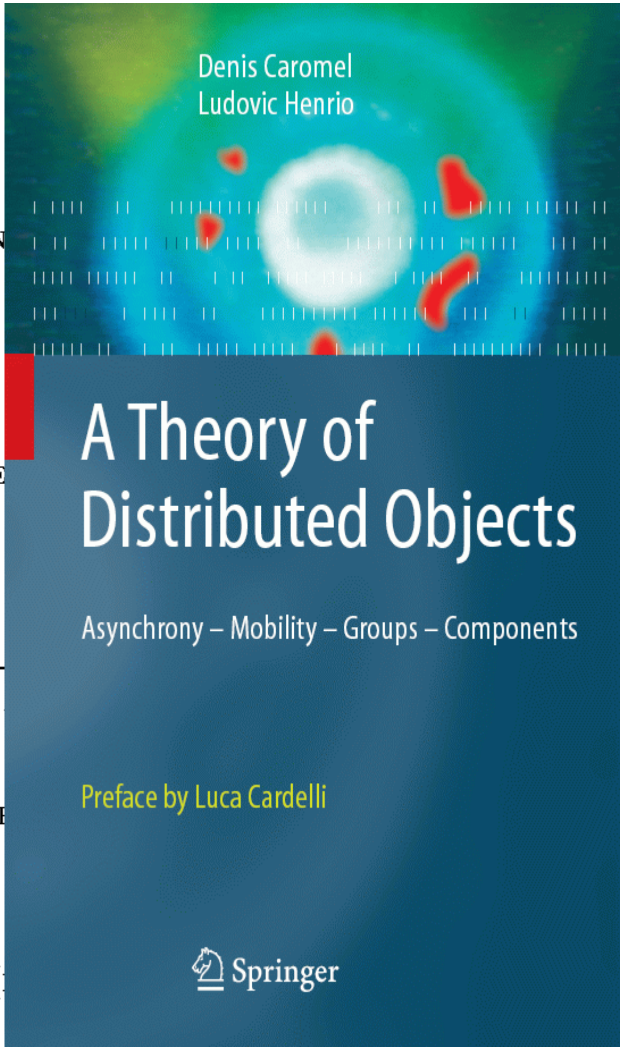
$$\frac{\begin{array}{c}\gamma\text{ fresh activity}\qquad \iota'\notin dom(\sigma)\qquad \sigma'=\{\iota'\mapsto AO(\gamma)\}::\sigma\\ \sigma_\gamma=copy(\iota'',\sigma)\qquad Service=(\text{ if }m_j=\emptyset\text{ then }FifoService\text{ else }\iota''.m_j())\end{array}}{\begin{array}{c}\alpha[\mathcal{R}[Active(\iota'',m_j)];\sigma;\iota;F;R;f]\parallel P\\ \longrightarrow\alpha[\mathcal{R}[\iota'];\sigma';\iota;F;R;f]\parallel\gamma[Service;\sigma_\gamma;\iota'';\emptyset;\emptyset;\emptyset]\parallel P\end{array}}\ (\text{N}$$

$$\frac{\begin{array}{c}\sigma_\alpha(\iota)=AO(\beta)\qquad \iota''\notin dom(\sigma_\beta)\qquad f_i^{\alpha\to\beta}\text{ new future}\qquad \iota_f\notin dom(\sigma_\alpha)\\ \sigma'_\beta=Copy\&Merge(\sigma_\alpha,\iota'\ ;\ \sigma_\beta,\iota'')\qquad \sigma'_\alpha=\{\iota_f\mapsto fut(f_i^{\alpha\to\beta})\}::\sigma_\alpha\end{array}}{\begin{array}{c}\alpha[\mathcal{R}[\iota.m_j(\iota')];\sigma_\alpha;\iota_\alpha;F_\alpha;R_\alpha;f_\alpha]\parallel\beta[a_\beta;\sigma_\beta;\iota_\beta;F_\beta;R_\beta;f_\beta]\parallel P\longrightarrow\\ \alpha[\mathcal{R}[\iota_f];\sigma'_\alpha;\iota_\alpha;F_\alpha;R_\alpha;f_\alpha]\parallel\beta[a_\beta;\sigma'_\beta;\iota_\beta;F_\beta;R_\beta::[m_j;\iota'';f_i^{\alpha\to\beta}];f_\beta]\parallel P\end{array}}\ (\text{RE}$$

$$\frac{R=R'::[m_j;\iota_r;f']::R''\qquad m_j\in M\qquad \forall m\in M,\ m\notin R'}{\alpha[\mathcal{R}[Serve(M)];\sigma;\iota;F;R;f]\parallel P\longrightarrow\alpha[\iota.m_j(\iota_r)\Uparrow f,\mathcal{R}[[]];\sigma;\iota;F;R'::R'';f']\parallel}$$

$$\frac{\iota'\notin dom(\sigma)\qquad F'=F::\{f\mapsto\iota'\}\qquad \sigma'=Copy\&Merge(\sigma,\iota\ ;\ \sigma,\iota')}{\alpha[\iota\Uparrow(f',a);\sigma;\iota;F;R;f]\parallel P\longrightarrow\alpha[a;\sigma';\iota;F';R;f']\parallel P}\ (\text{ENDS}$$

$$\frac{\sigma_\alpha(\iota)=fut(f_i^{\gamma\to\beta})\qquad F_\beta(f_i^{\gamma\to\beta})=\iota_f\qquad \sigma'_\alpha=Copy\&Merge(\sigma_\beta,\iota_f\ ;\ \sigma_\alpha,\iota)}{\begin{array}{c}\alpha[a_\alpha;\sigma_\alpha;\iota_\alpha;F_\alpha;R_\alpha;f_\alpha]\parallel\beta[a_\beta;\sigma_\beta;\iota_\beta;F_\beta;R_\beta;f_\beta]\parallel P\longrightarrow\\ \alpha[a_\alpha;\sigma'_\alpha;\iota_\alpha;F_\alpha;R_\alpha;f_\alpha]\parallel\beta[a_\beta;\sigma_\beta;\iota_\beta;F_\beta;R_\beta;f_\beta]\parallel P\end{array}}\ ($$



ASP $\Rightarrow$ Confluence and Determinacy

Future updates can occur at any time, Mobility does not change behavior

TYPED

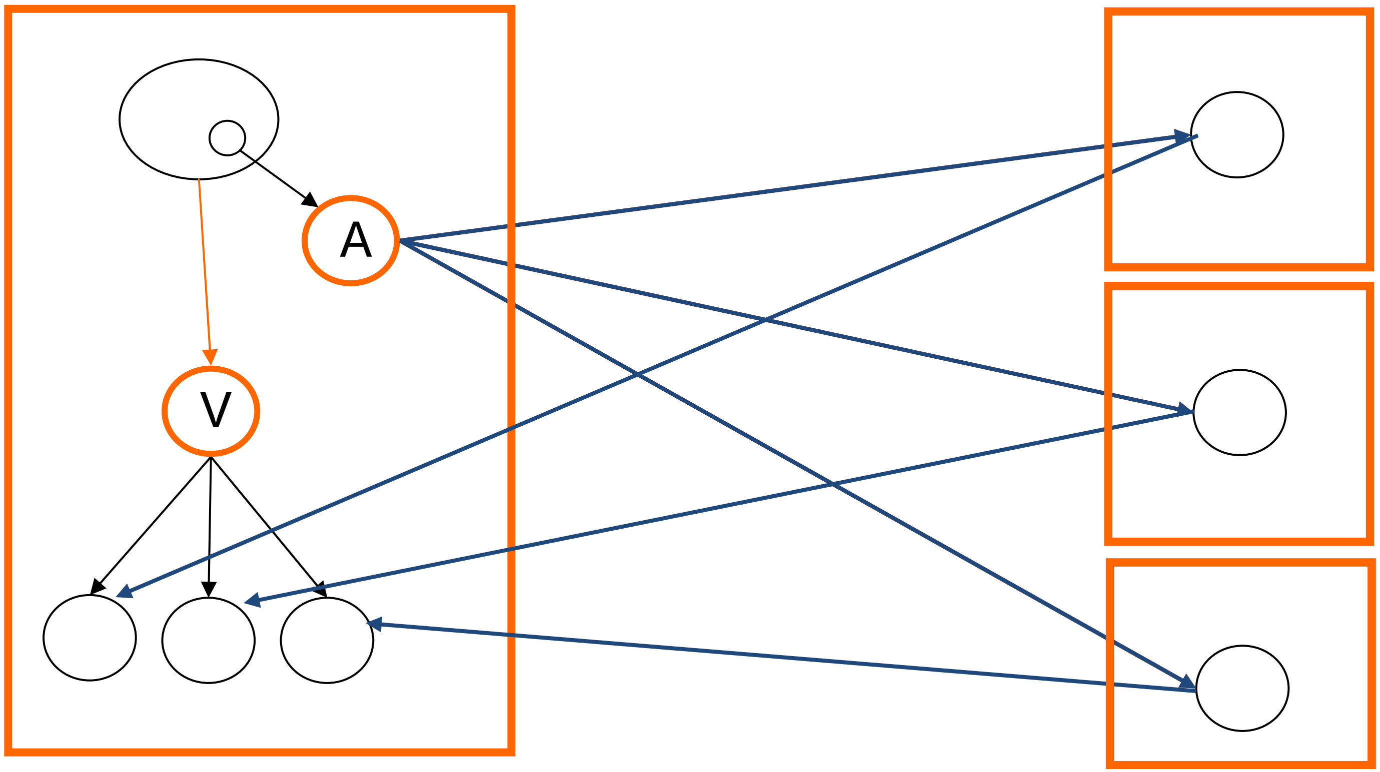
ASYNCHRONOUS

GROUPS

Creating AO and Groups

- `A ag = newActiveGroup (“A”, […], VirtualNode)`
- `V v = ag.foo(param);`
- ...
- `v.bar(); //Wait-by-necessity`

JVM

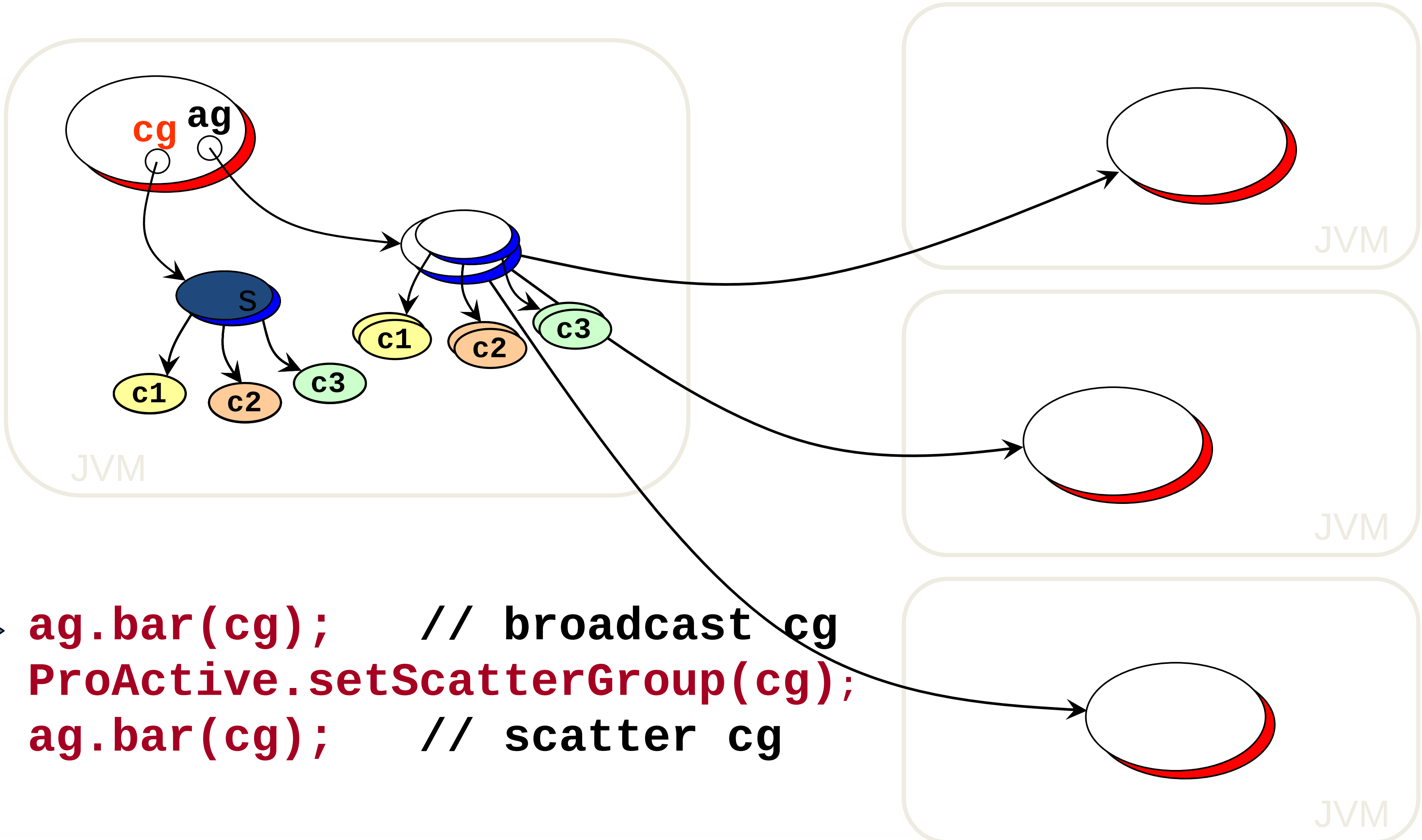


Typed Group Java or Active Object

Group, Type, and Asynchrony are crucial for Cpt. and GRID

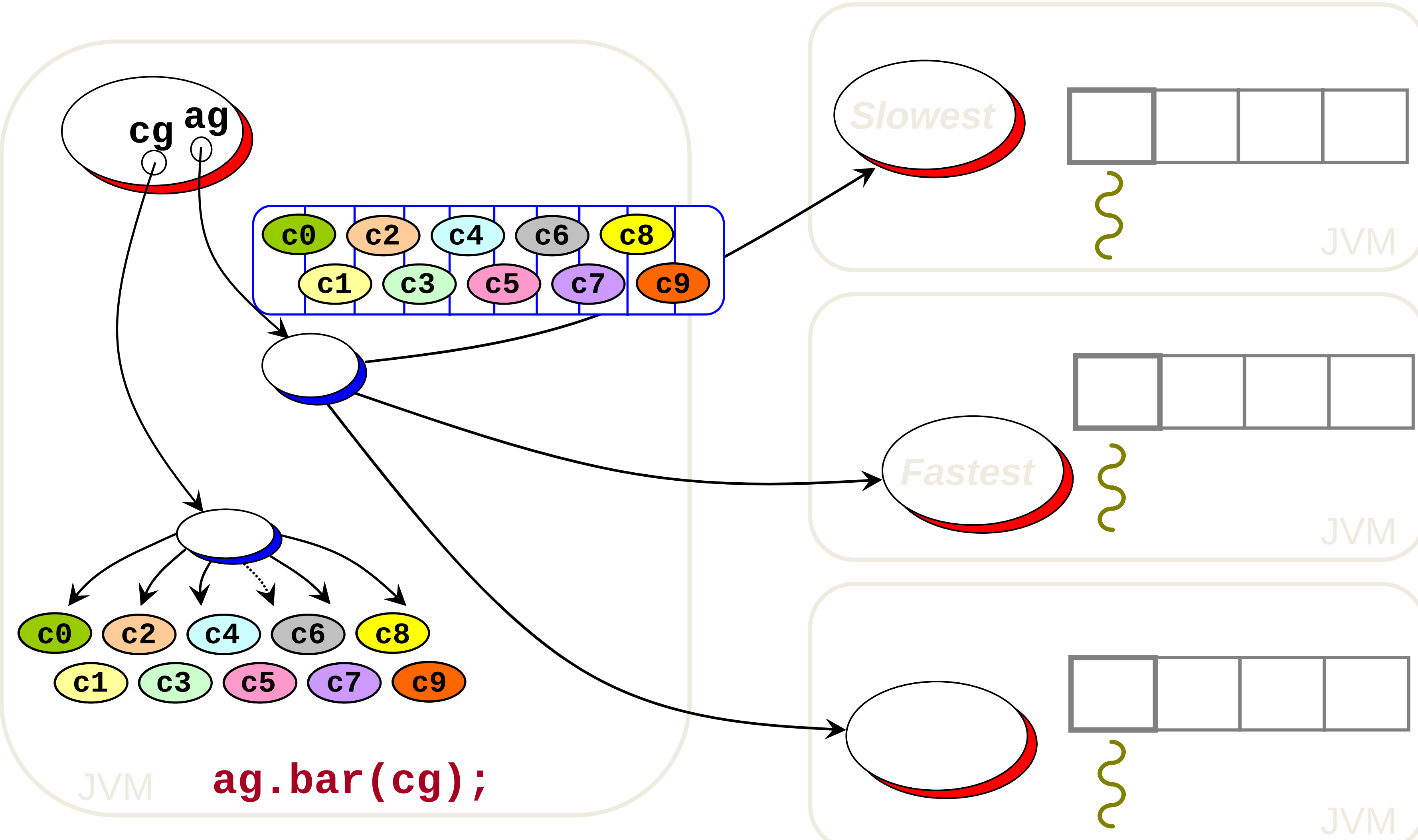
Broadcast and Scatter

Broadcast is the default behavior
Use a group as parameter, Scattered depends on rankings




```
ag.bar(cg);   // broadcast cg
ProActive.setScatterGroup(cg);
ag.bar(cg);   // scatter cg
```

Dynamic Dispatch Group

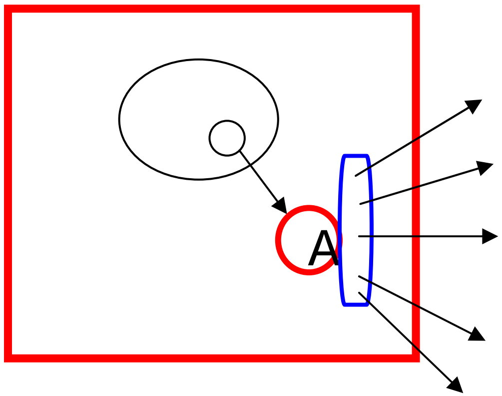


OO SPMD

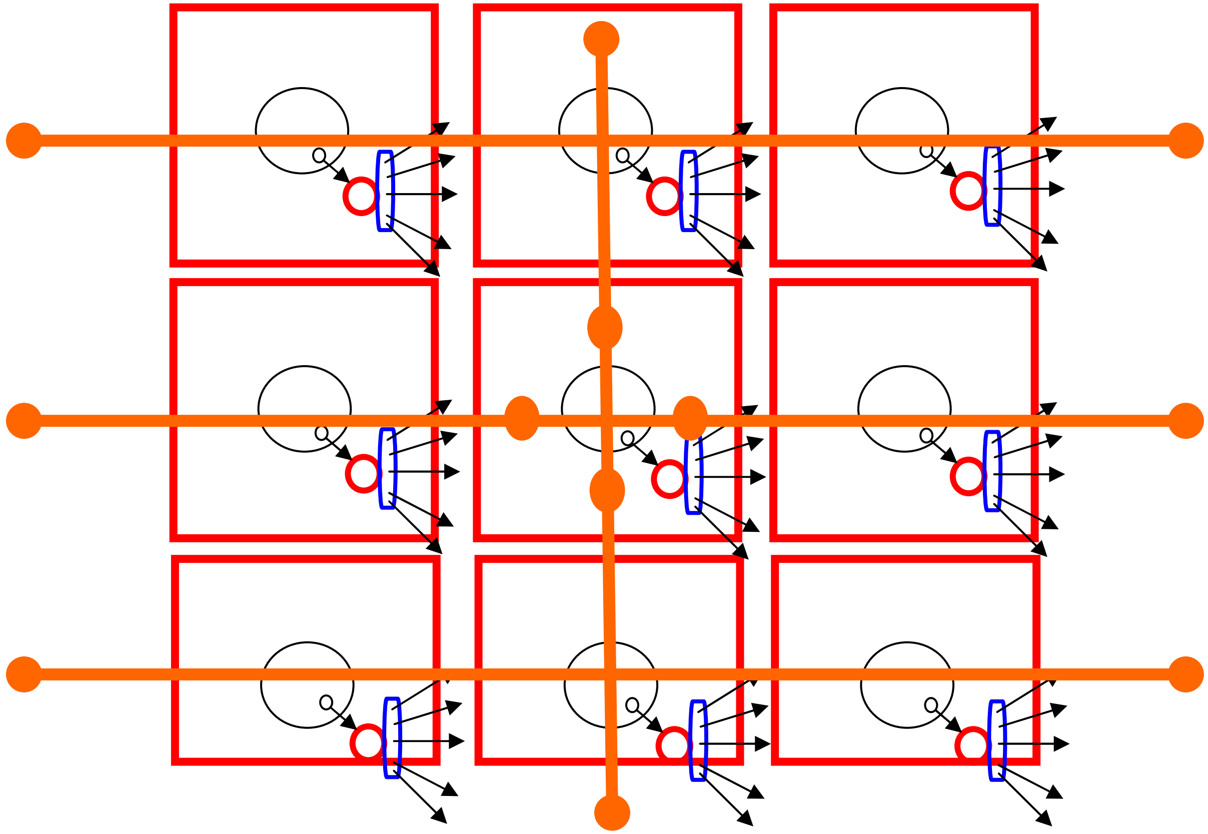
- `A ag = newSPMDGroup ("A", […], VirtualNode)`

```
// In each member
```

- `myGroup.barrier ("2D"); // Global Barrier`
- `myGroup.barrier ("vertical"); // Any Barrier`
- `myGroup.barrier ("north","south","east","west");`



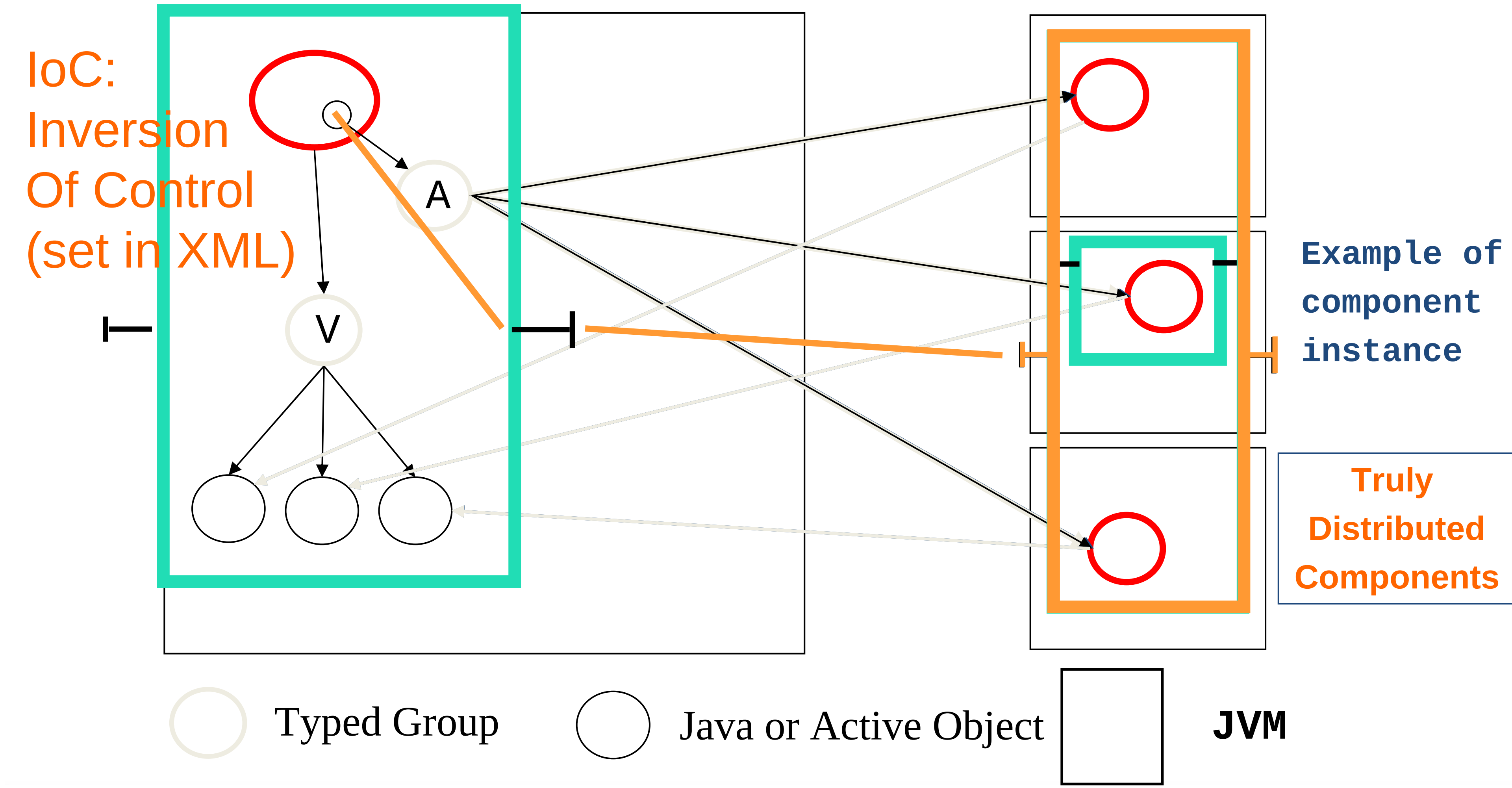
Still,
not based on raw messages, but
Typed Method Calls
==> Components



3. Components and Standardization (GCM)

Objects to Distributed Components (1)

```
ComponentIdentity Cpt = newActiveComponent (params);
A a = Cpt … .getFcInterface ("interfaceName");
V v = a.foo(param);
```



GCM: Grid Component Model

- GCM Being defined in the NoE CoreGRID
  - (42 institutions)
- Open Source ObjectWeb *ProActive*
  - implements a preliminary version of GCM
- Service Oriented: NESSI relation

The vision: GCM to be the IT Service GSM

GridCOMP takes:

- GCM as a first specification,
- *ProActive* as a starting point, and
  - Open Source reference implementation.

GridCOMP Partners

Sco

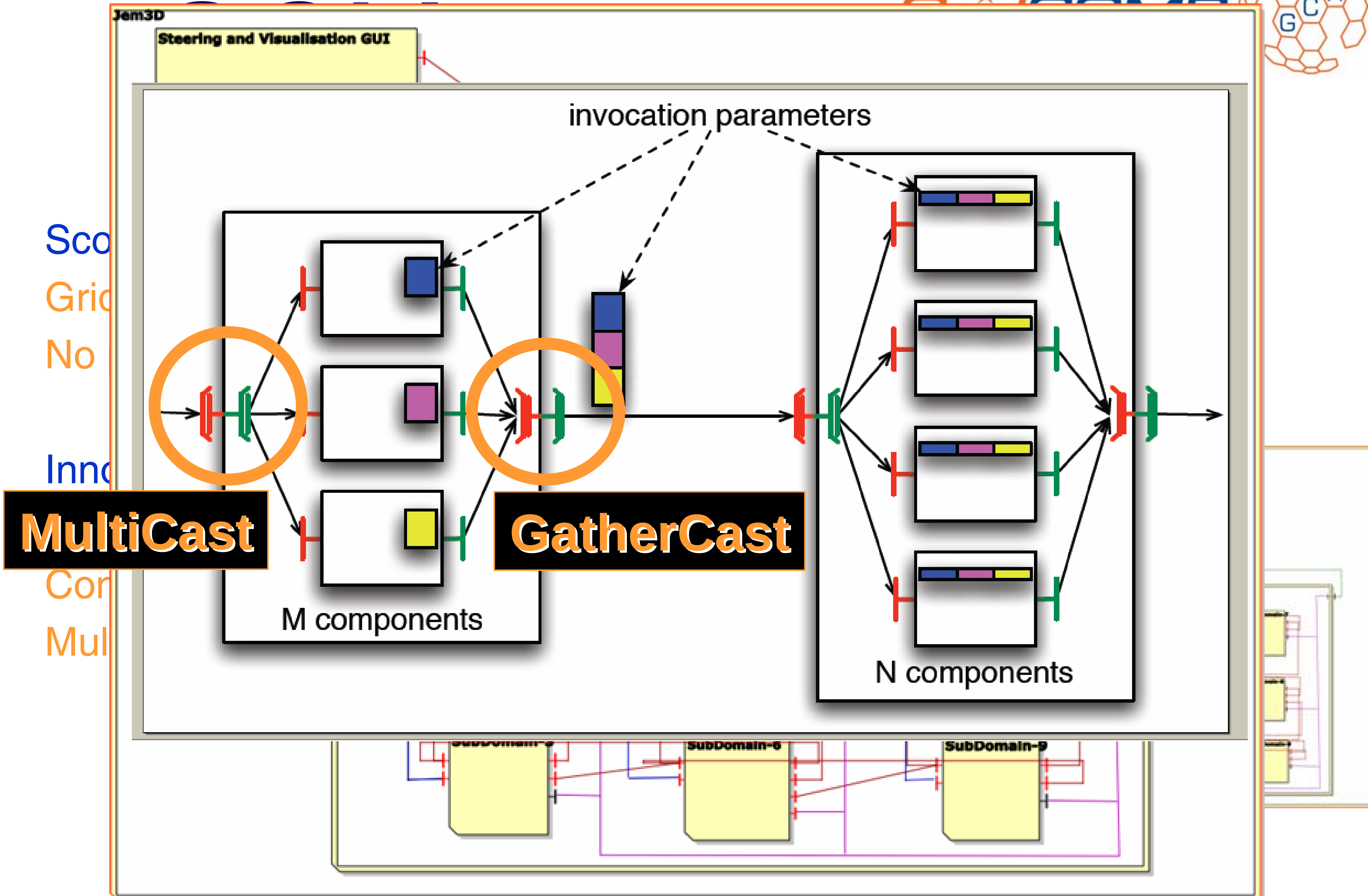
Grid

No

Inno

Con

Mul



Standardization

ETSI

World Class Standards

GCM ETSI STANDARDIZATION

ETSI TC GRID Standardization Group :

one meeting every 3 or 4 months since Oct. 2006

On 12 June 2008, at the #8 ETSI TC Grid meeting, the two standards:

GCM Interoperability Deployment

GCM Interoperability Application Description

have been officially approved!

Overall, the standardization is supported by industrials:

BT, FT-Orange, Nokia-Siemens, Telefonica, NEC, Alcatel-Lucent, Huawei …

ETSI GCM TC Grid Standard

Official Standard No 1

GCM Interoperability Deployment

Official Standard No 2

GCM Application Description

Work Item No 3

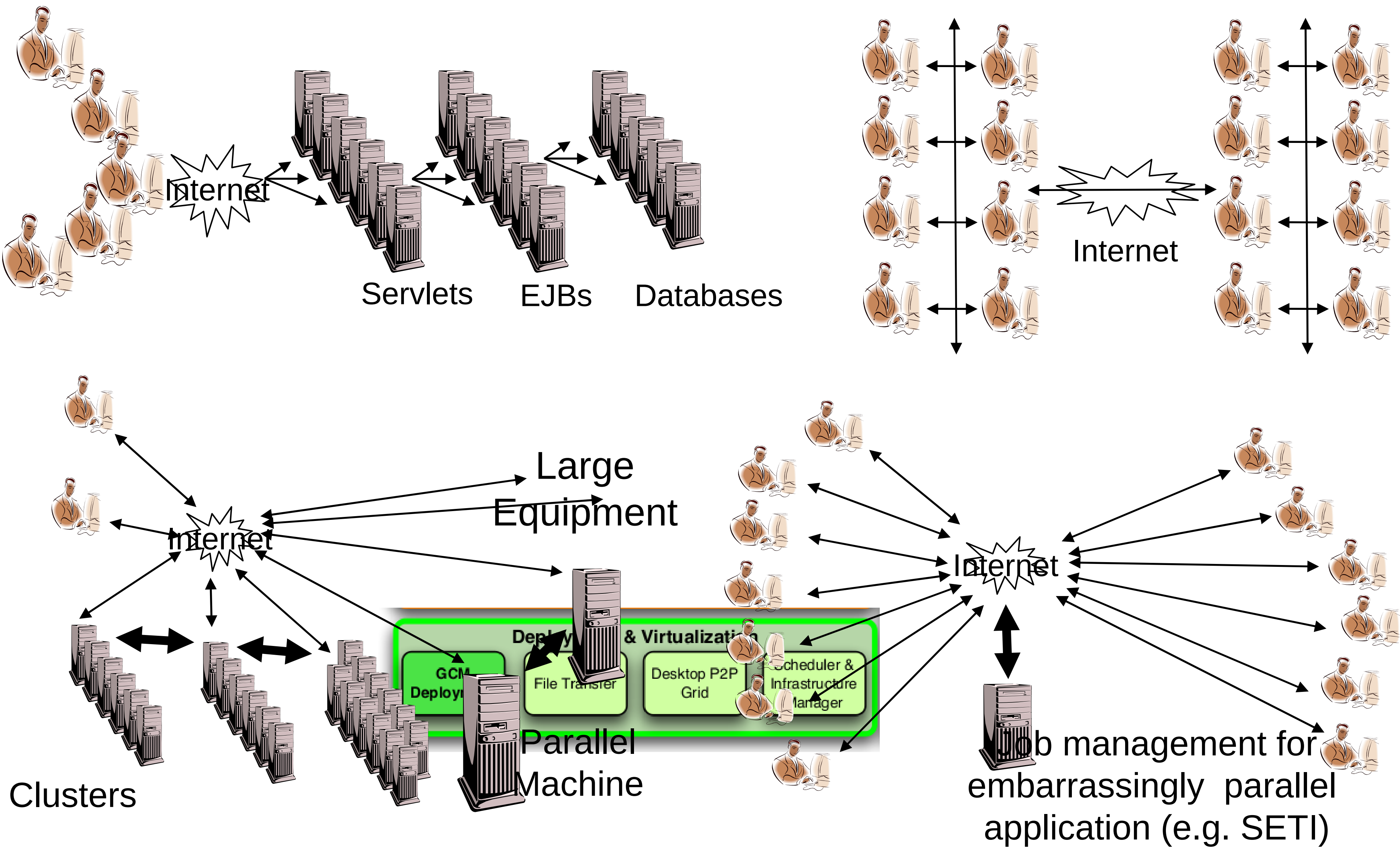
GCM Fractal ADL

(Architecture Description Language)

Work Item No 4

GCM Management (Java, C, WSDL API)

Deploy on Various Kinds of Infrastructures



Protocols and Scheduler in GCM Deployment

Protocols:

- rsh
- ssh
- Oarsh
- Gsissh

Scheduler, and Grids:

- GroupSSH, GroupRSH, GroupOARSH
- ARC (NorduGrid), CGSP China Grid, EEGE gLITE,
- Fura/InnerGrid (GridSystem Inc.)
- GLOBUS
- GridBus
- IBM Load Leveler, LSF, Microsoft CCS (WHPC 2008)
- Sun Grid Engine, OAR, PBS / Torque, PRUN

GridCOMP / GCM ProActive Usage

Used in Production by Companies:

- E.g. Amadeus (Air France, Lufthansa)

At least 4 on going PhD. thesis:

- Component reconfiguration (Marcela Rivera),
- GCM extensions for autonomic applications (Paul Naoumenko),
- Specification Languages and Model-Checking (Antonio Cansado)
- Autonomic Service Management of Enterprise Grid Services (Cristian Ruz)

Used in other projects:

- EU PROJECTS: SOA4ALL, QosCosGrid, Prospect: RESERVOIR
- INRIA ADT Galaxy, Pole Comp. AGOS (HP, Oracle)

Used in:

- Barcelona
- Krakow (SemMon: Semantic Monitoring)
- …

4.

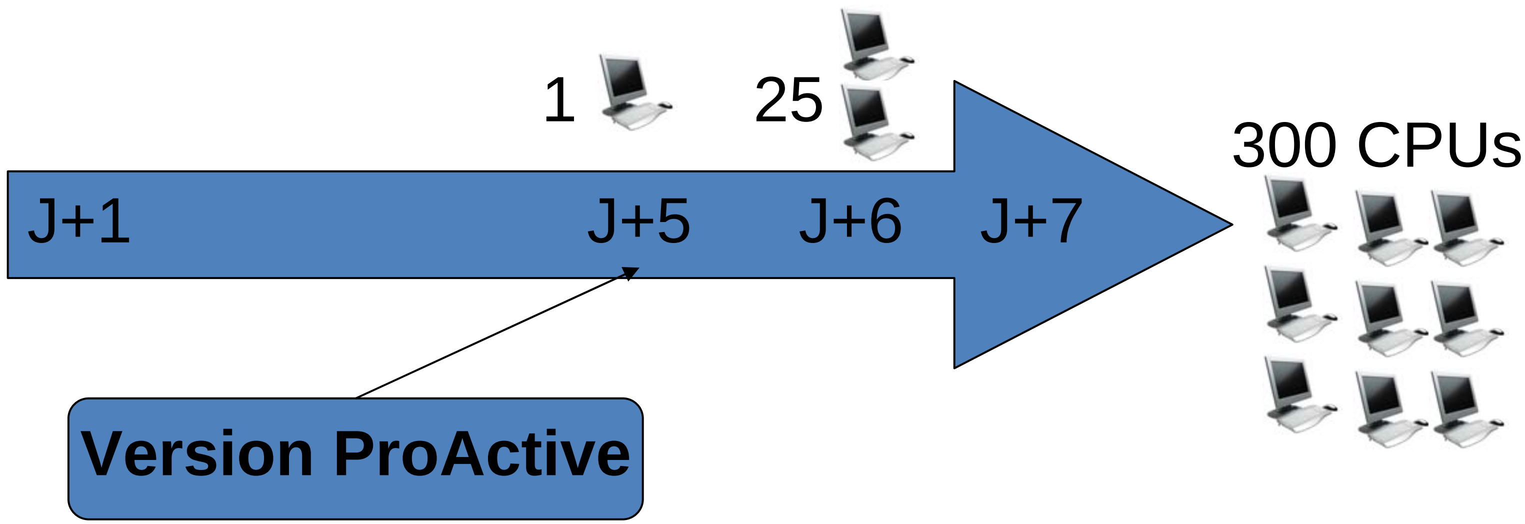
Applications and Perspectives: SOA+GRID

Artificial Life Generation

Sylvain Cussat-Blanc, Yves Duthen – IRIT TOULOUSE

Application

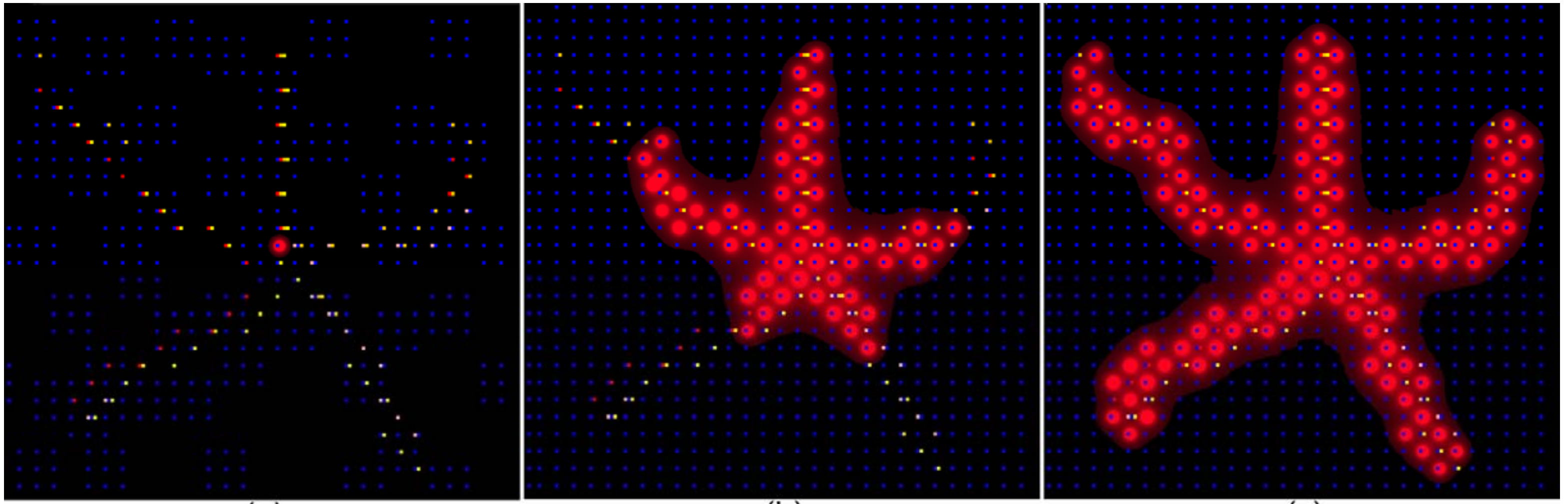
Developpement of artificial creatures



Initial Application	**1 PC**	**56h52 => Crash!**
ProActive Version	**300 CPUs**	**19 minutes**

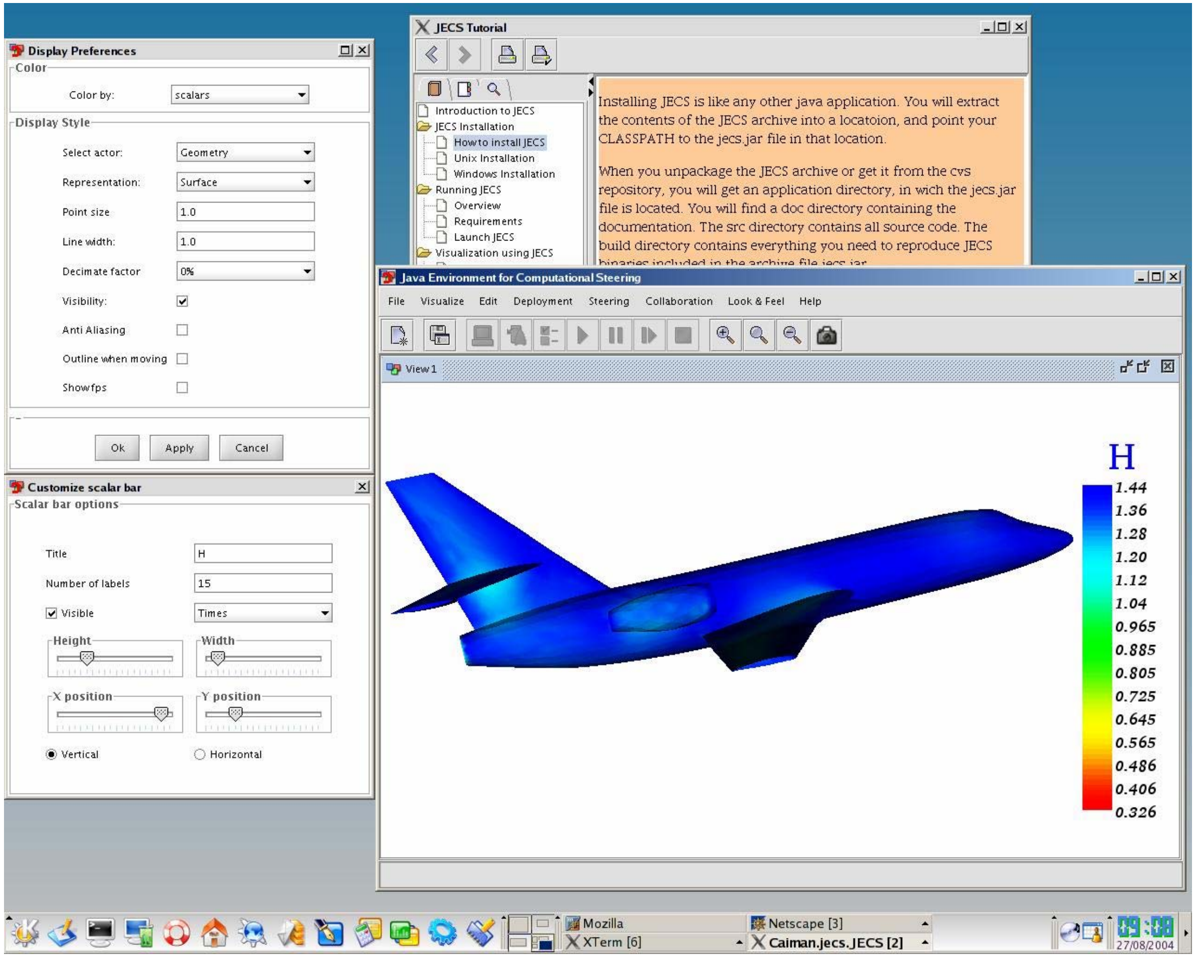
Artificial Life Generation

Sylvain Cussat-Blanc, Yves Duthen – IRIT TOULOUSE

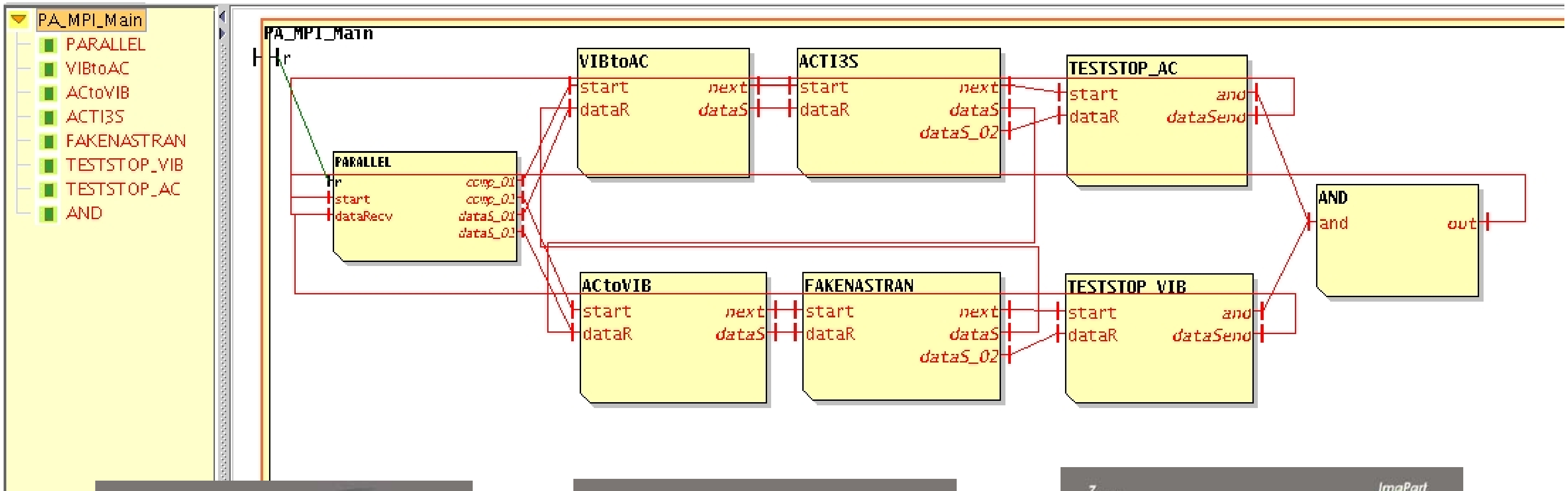


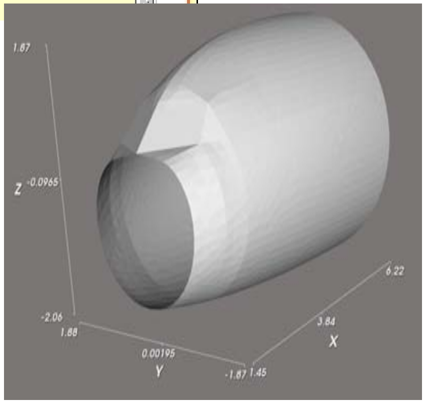
(a) (b) (c)

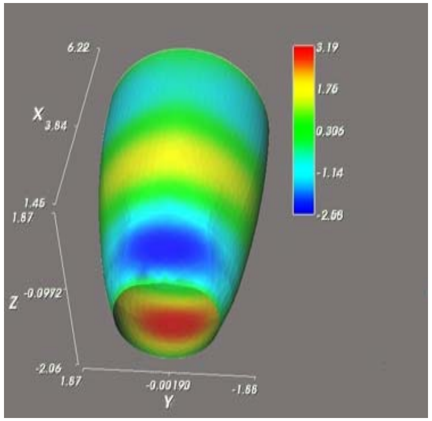
JECS : 3D Electromagnetism Radar Reflection on Planes

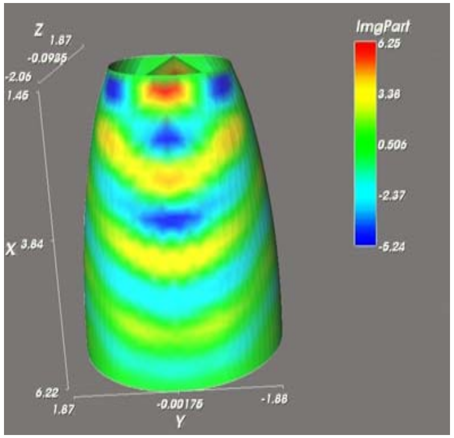


Code Coupling :
Vibro Acoustic (courtesy of EADS)



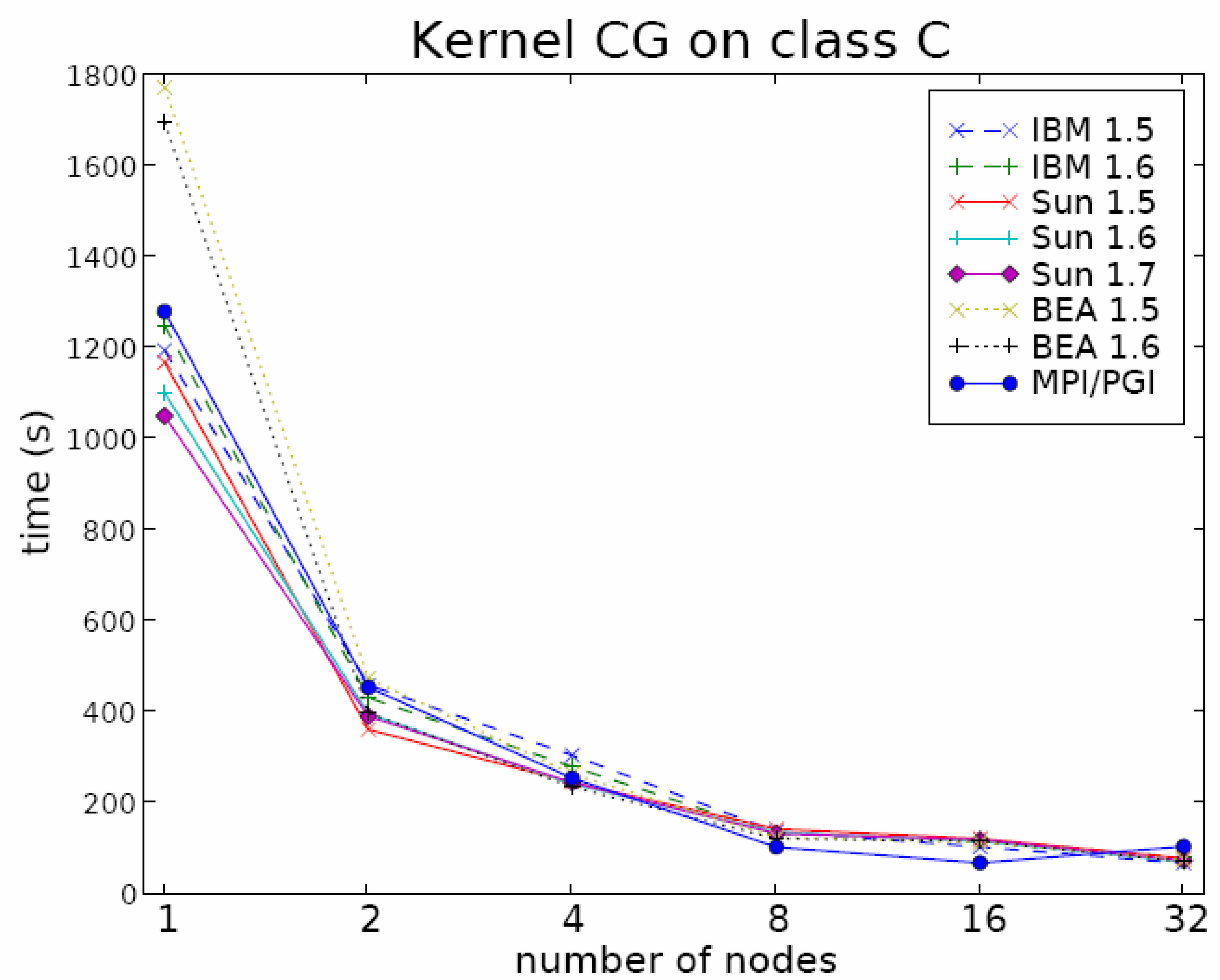






NAS Parallel Benchmarks

- *Desig*
  sy
- *Stron*
- *5 ben*
- *2 cate*
  int



Enterprise IT: Software Tests

Amadeus (Opodo, Air France, KLM, Lufthansa):

500 programmers → 20 machines with

ProActive to execute

Dist. Regression Tests

in the production chain

Parallel BLAST with ProActive (1)
together with Mario Leyton

Basic **L**ocal **A**lignment **S**earch **T**ool for rapid sequence comparison

BLAST developed by NCBI (**N**ational **C**enter for **B**iotechnology **I**nformation)

Standard native code package, no source modification!

With *PPS Skeletons* parallelization and distribution *added to the application*

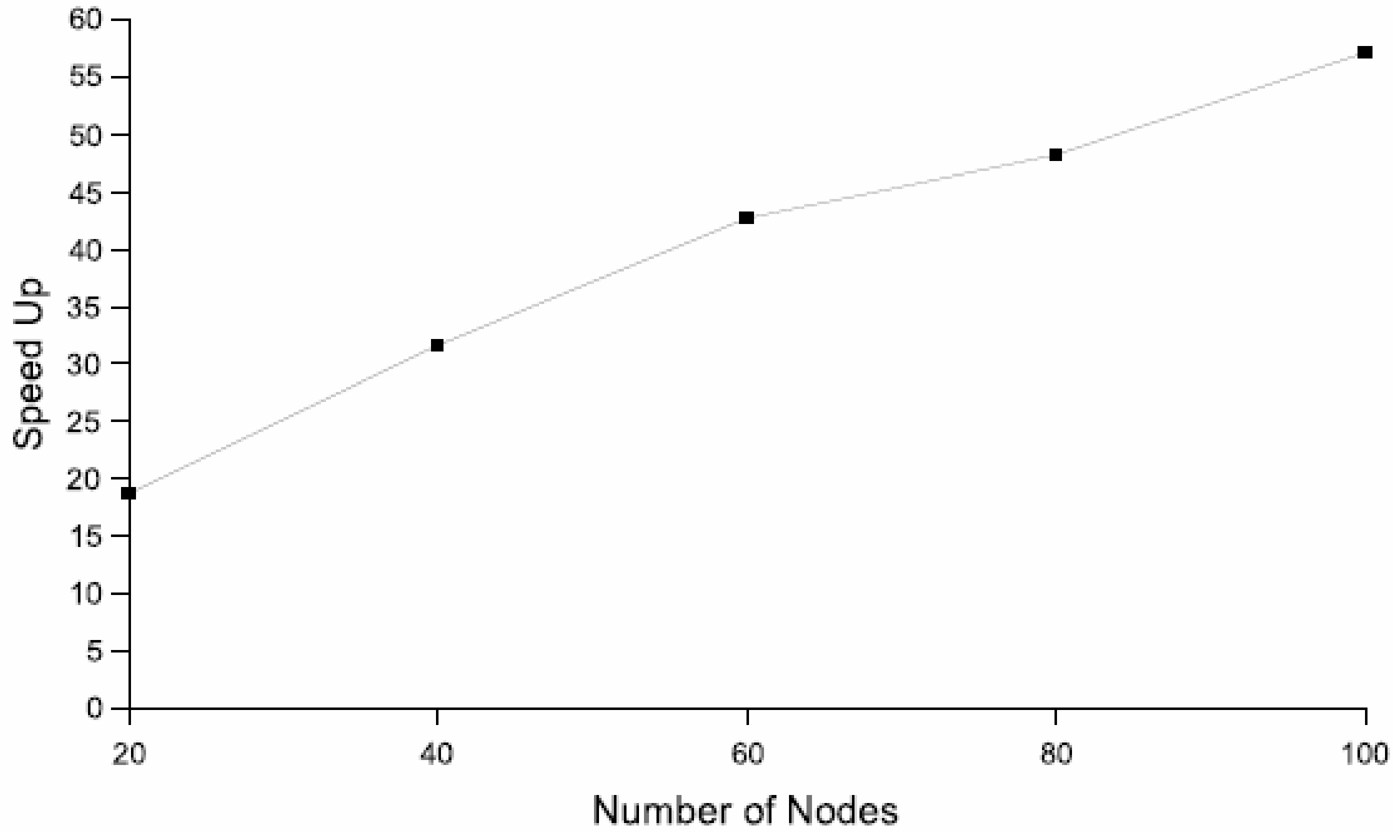
A seamless deployment on all Grid platforms is obtained:

- Input Files are automatically copied to computational nodes at Job submission
- Result Files will be copied on client host

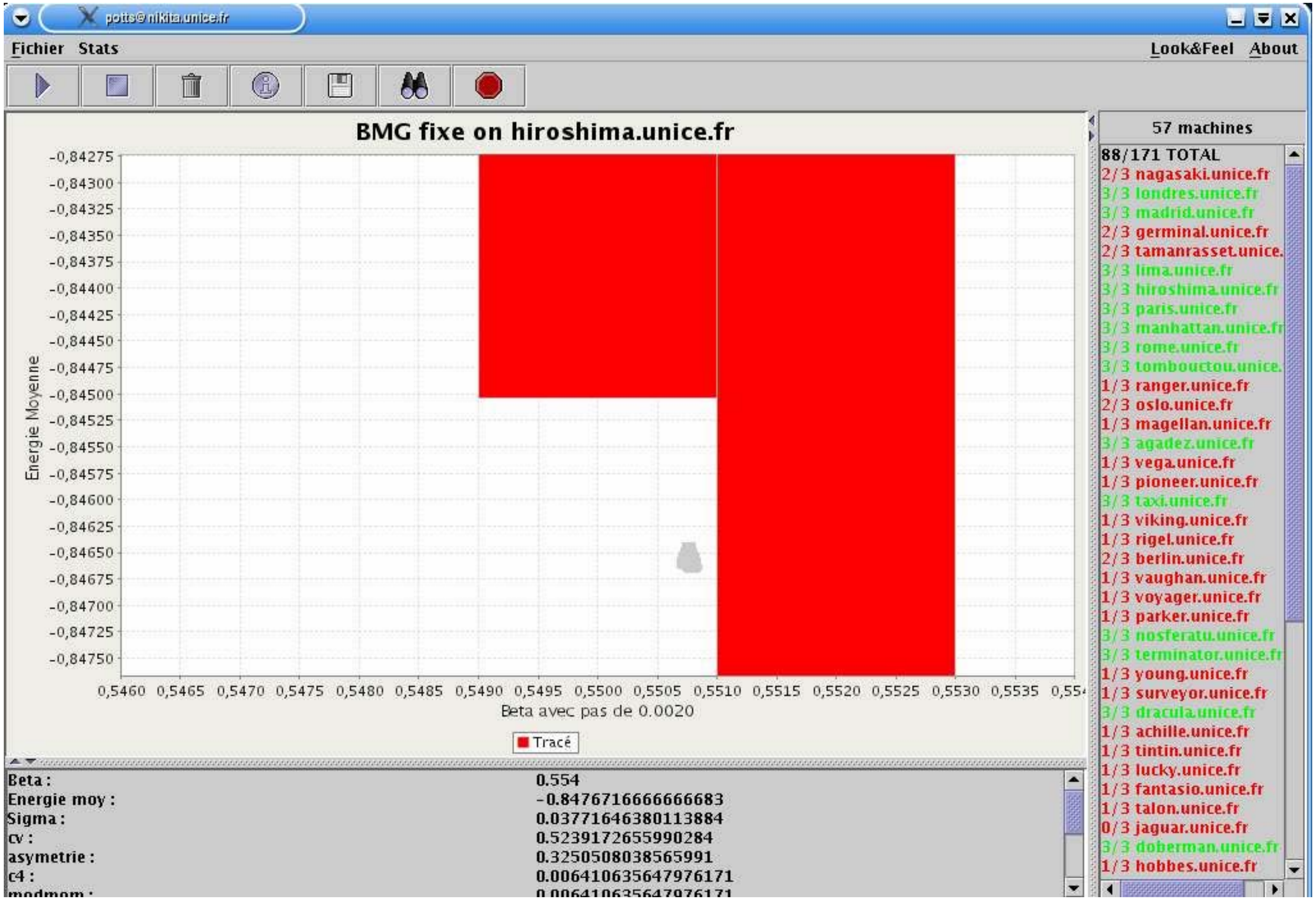
BLAST Skeleton program using the Divide and Conquer skeleton:

- Division of Database based on conditions (Nb. Nodes, Size, etc.)

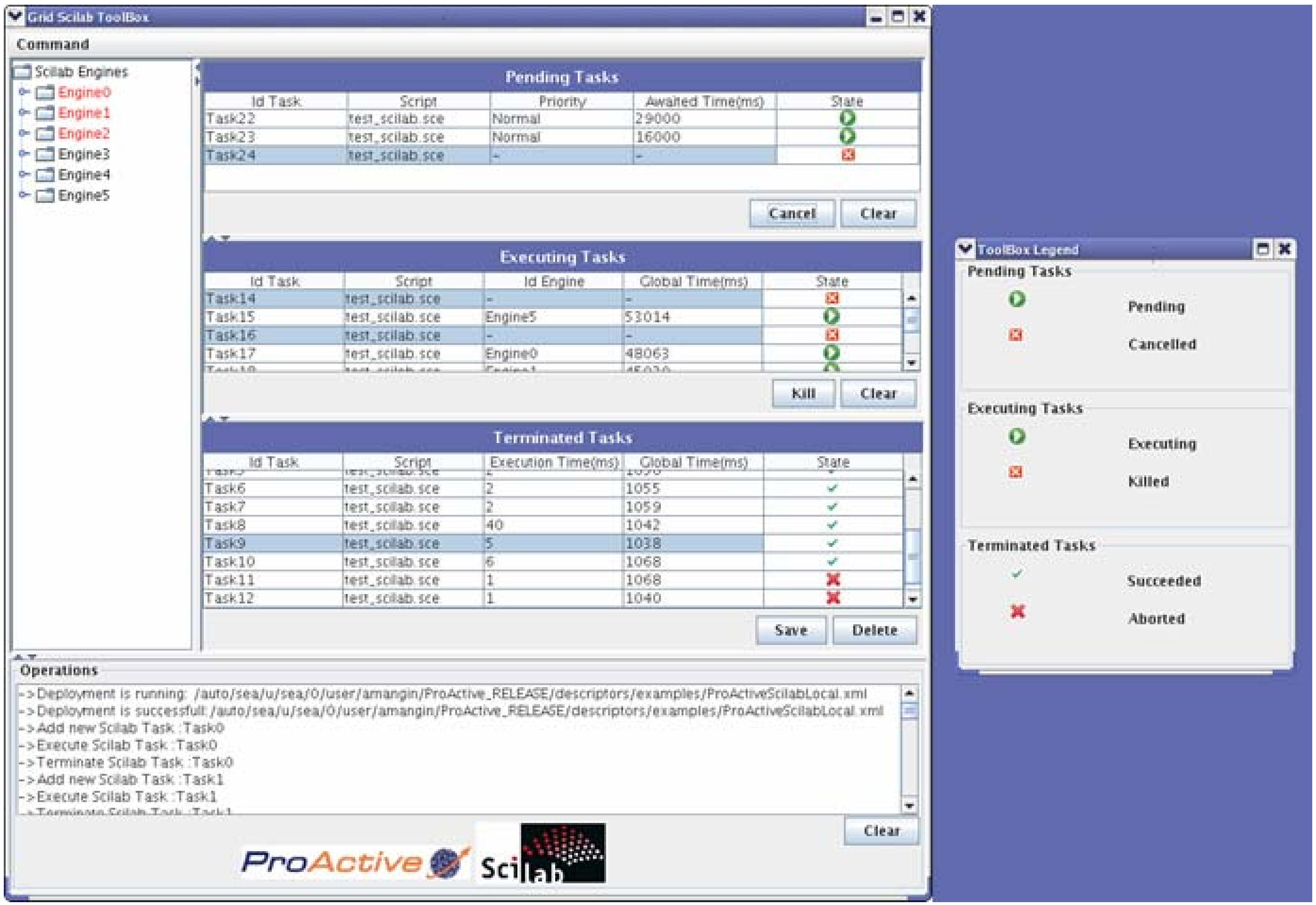
Speedup of Distributed BLAST on Grid5000



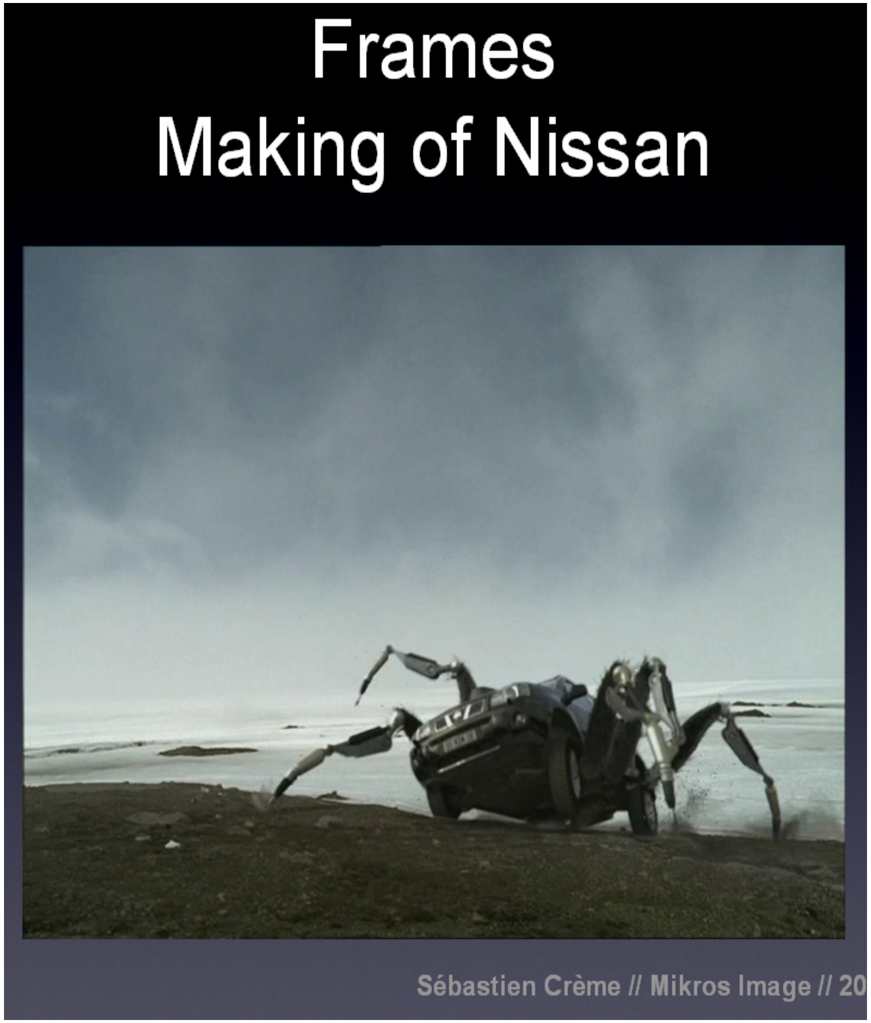
Monte Carlo Simulations, Non-Linear Physics, INLN



Matlab and Scilab Grid Interface



Mikros Image: Post Production



New Developments:

Grid & SOA

AGOS
Grid Architecture for SOA
Building a Platform for Agile SOA with Grid

Partners and Solutions

Use Cases

AGOS: What for ?

AGOS Objectives:

- Create an architecture and environment for integration of
  - SOA business management with
  - GRID IT management
- Well fitted for data intensive and computational intensive applications:
  - Enact sub-parts of a BPEL workflow on dynamically allocated resource
    E.g.: Financial Simulations, Insurance, Revenue Management, BIO, HPC
- Full dynamic scheduling of Services on GRIDs in the future
- Integrated Management of SLO, SLA, QoS:
  - Bottom to top
  - Dynamic enforcement: Adaptive behavior

Summary

ProActive Parallel Suite

Concurrency + Parallelism
Multi-Cores + Distribution

Conclusion: Why does it scale?

Thanks to a few key features:

Connection-less, RMI+JMS unified

Messages rather than long-living interactions

ACTIVE OBJECTS --- GROUPS --- COMPONENTS

Conclusion:
Why does it Compose?

Thanks to a few key features:

Because it Scales: asynchrony !

Because it is Typed: RMI with interfaces !

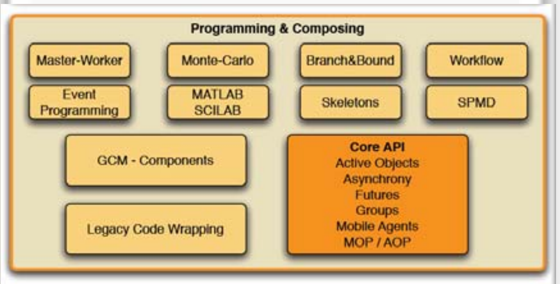
First-Class Futures: No unstructured Call Backs and Ports

ACTIVE OBJECTS --- GROUPS --- COMPONENTS

Conclusion:

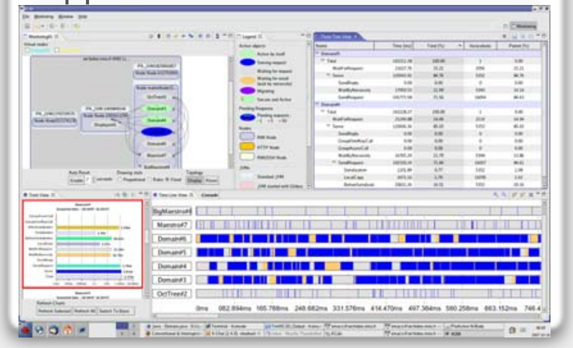
PROGRAMMING

Java Parallel Frameworks
for HPC, Multi-Cores, Distribution, Enterprise Grids and Clouds.



OPTIMIZING

Eclipse GUI (IC2D)
for Developing, Debugging, Optimizing your parallel applications.



SCHEDULING

Multi-Language Scheduler
for Workflows made of C, C++, Java, Scripts, Matlab, Scilab tasks.

A Toolkit for Acceleration:

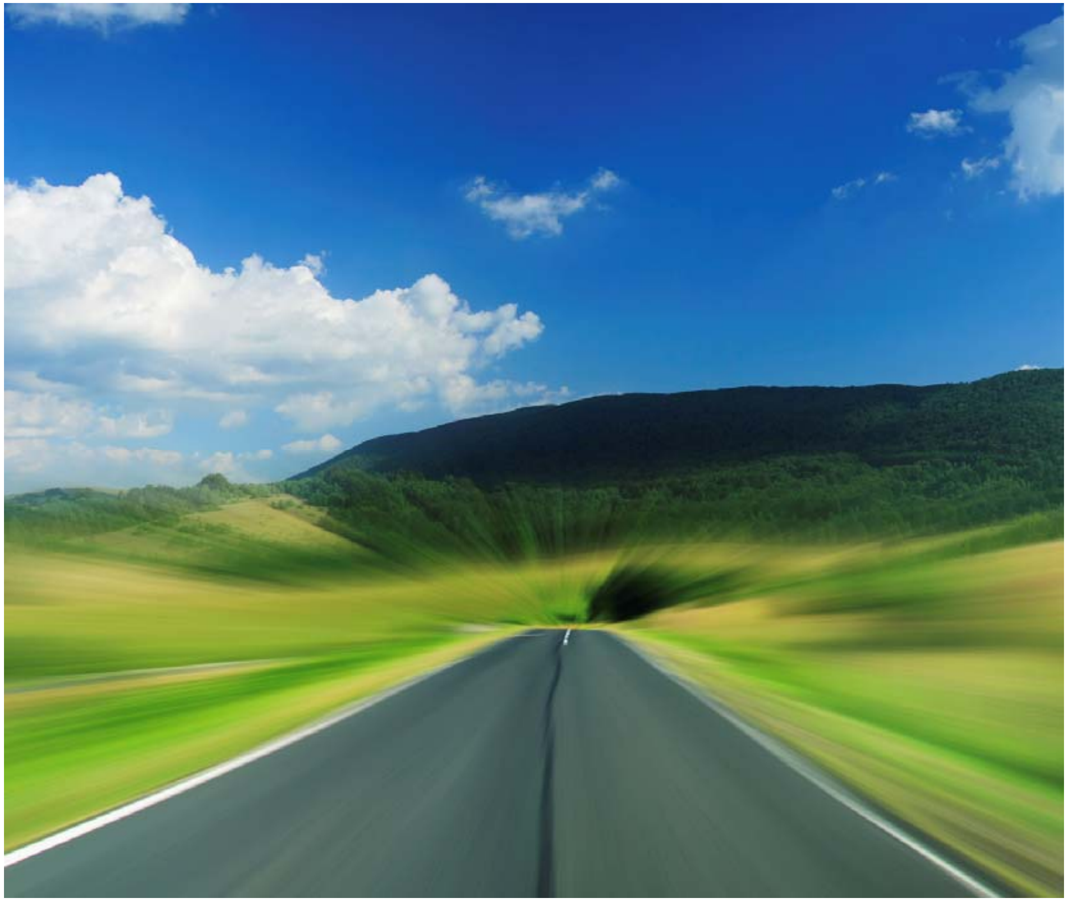
Multi-Core & Distributed

ProActive/ GCM Specifications

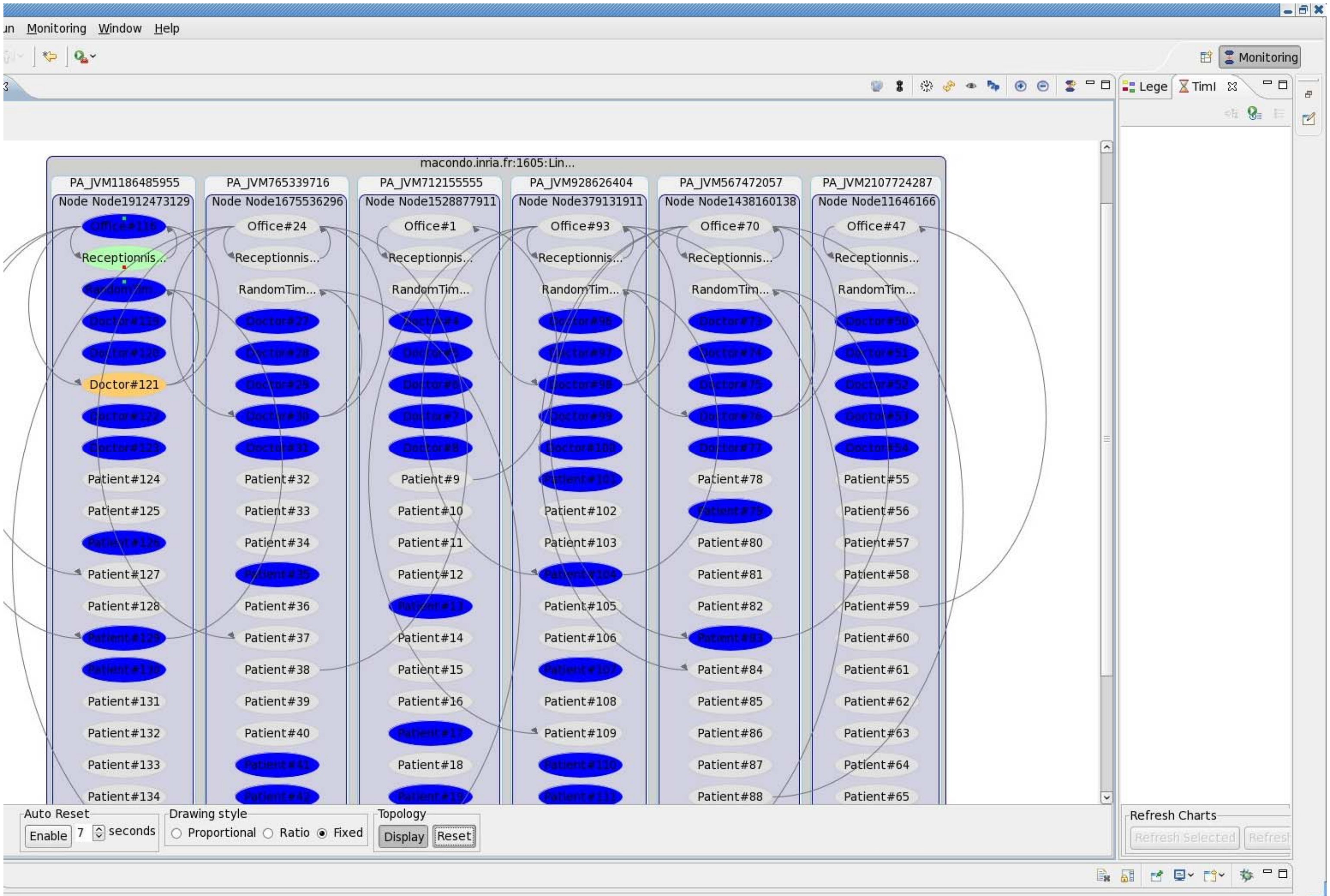
for

Components
Services
SLA
QoS

Open the way to Soft.+Serv. EU Industry with Clouds & Utilities, DAAS



Multi-Active Object in 1 Address Space for Multi-cores



ProActive Parallel Suite

search

ProActive Parallel Suite

Professional Open Source Middleware
for Parallel, Distributed, Multi-core Programming

Home | Services | Documentation | Downloads | Community | About us

› Home

Trainings

ProActive Overview and Application to Finance
Location: French Riviera
Schedule:
- 22, 23rd October 2008

News

NEW TECHNICAL PAPER:
Current State of Java for HPC

5th Grid Plugtests (Grids@Work)
20-24 October 2008
Monte-Carlo Finance Contest and **ProActive User Group**

Invited Keynote at **CGW'08**, Kraków, Poland
October, 13-15, 2008
Grid Component Model and ProActive Parallel Suite for Science

Careers

Engineer positions available, click **here** to learn more.

Latest releases

Latest:
ProActive 4.0.1, Sep. 2008
[Download ProActive]

Welcome to the Open Source World of ProActive !

New ! ProActive 4.0.1 (15/09/2008)

ProActive Parallel Suite is an **Open Source** middleware (**OW2** consortium) for **parallel**, **distributed**, **multi-core** computing.

ProActive tools **simplify** the programming and execution of parallel applications on: Multi-core processors, Distributed Local Area Network (LAN), Clusters and Data Center Servers, GRIDs, and on OS such as Linux, Windows, Mac.

ProActive Parallel Suite features:

PROGRAMMING

Java Parallel Frameworks
for HPC, Multi-Cores, Distribution, Enterprise Grids and Clouds.

Featuring: Async. comms, Master-Worker, Monte-Carlo, SPMD, components and legacy code wrapping.

OPTIMIZING

Eclipse GUI (IC2D)
for Developing, Debugging, Optimizing your parallel applications.

Featuring: graphical monitoring and benchmarking with report generation.

SCHEDULING

Multi-Language Scheduler
for Workflows made of C, C++, Java, Scripts, Matlab, Scilab tasks.

Featuring: graphical user interface, resource acquisition and virtualization.

Featuring **fault-tolerance, load-balancing, mobility,** and **security,** ProActive brings ease-of-use in the world of parallel computing. Programmers write standard Java code (POJO) for their parallel and Grid applications.
The Grid Application Server for the Enterprise!

Developer Tools & Eclipse IDE Plugins

5 Minutes

Programming &

Professional

RAMMING

el
ks
ti-Cores,
 Enterprise
ouds.

sync. comms,
ker, Monte-
, components
ode wrapping.

OPTIMIZING

Eclipse GUI (IC2D)
for Developing, Debugging,
Optimizing your parallel
applications.

Featuring: graphical
monitoring and
benchmarking with report
generation.

SCHEDULI

Multi-Language Sc
for Workflows made
C++, Java, Scripts,
Scilab tasks.

Featuring: graphical
interface, resource a
and virtualization.

Object-Oriented SPMD
Single Program Multiple Data

Motivation

- Cluster / GRID computing
- SPMD programming for many numerical simulations
- **Use enterprise technology (Java, Eclipse, etc.) for Parallel Computing**

Able to express most of MPI's Collective Communications:

broadcast	**reduce**
scatter	**allscatter**
gather	**allgather**

and Barriers, Topologies.

GCM Deployment (2/2)

Grid description: clear concepts

- Bridges (1 -> 1)
- Groups (1 -> N)
- Hosts
- Acquisition (lookup, p2p)

Application description:

Split Grid / Application Description

Allows reuse of grid descriptors for any application type, ProActive, using Virtual nodes

Deployment descriptor : example

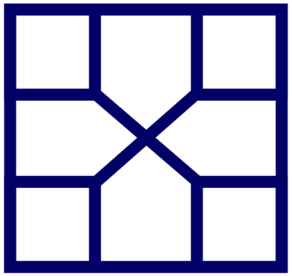
```xml
<resources>
    <bridge refid="bSchubby">
        <host refid="hSchubby" />
    </bridge>
</resources>

<acquisition>
    <lookup type="RMI" port="6666" hostList="host[0-9].grid.fr"></lookup>
    <p2p nodesAsked="50">
        <localClient protocol="RMI" port="2410" />
        <peerSet>
            <peer>rmi://schubby.inria.fr</peer>
            <peer>http://gaudi.inria.fr</peer>
        </peerSet>
    </p2p>
</acquisition>

<infrastructure>
    <hosts>
        <host id="hSchubby" os="unix" hostCapacity="1" vmCapacity="1">
            <homeDirectory base="root" relpath="/user/cmathieu/home" />
        </host>
    </hosts>

    <bridges>
        <sshBridge commandPath="/usr/bin/ssh" hostname="schubby.inria.fr" id="bSchubby" username="cmathieu" />

        <rshBridge hostname="schubby.inria.fr" id="brSchubby" username="cmathieu" />

    </bridges>
</infrastructure>
```

Communication Intensive
CG Kernel (Conjugate Gradient)

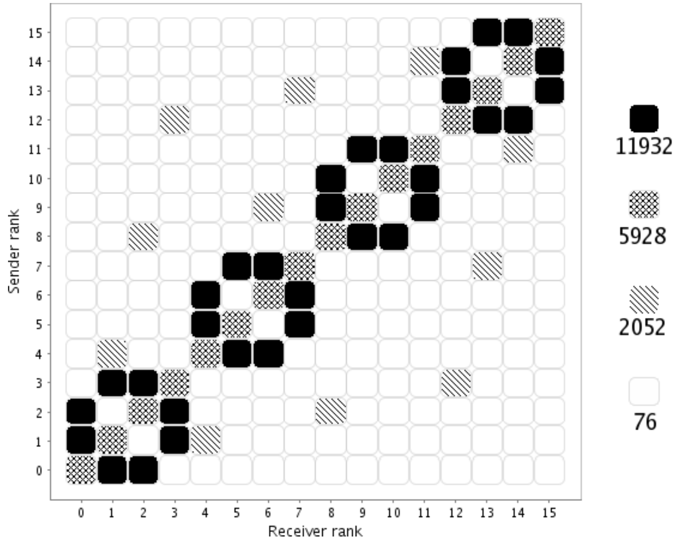
Floating point operations

Eigen value computation

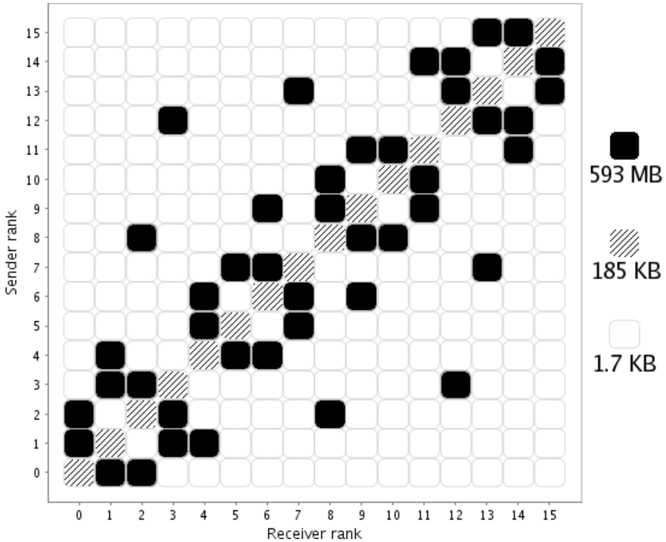
High number of unstructured communications



- 12000 calls
- 570 MB sent
- 1 min 32
- 65 % comms

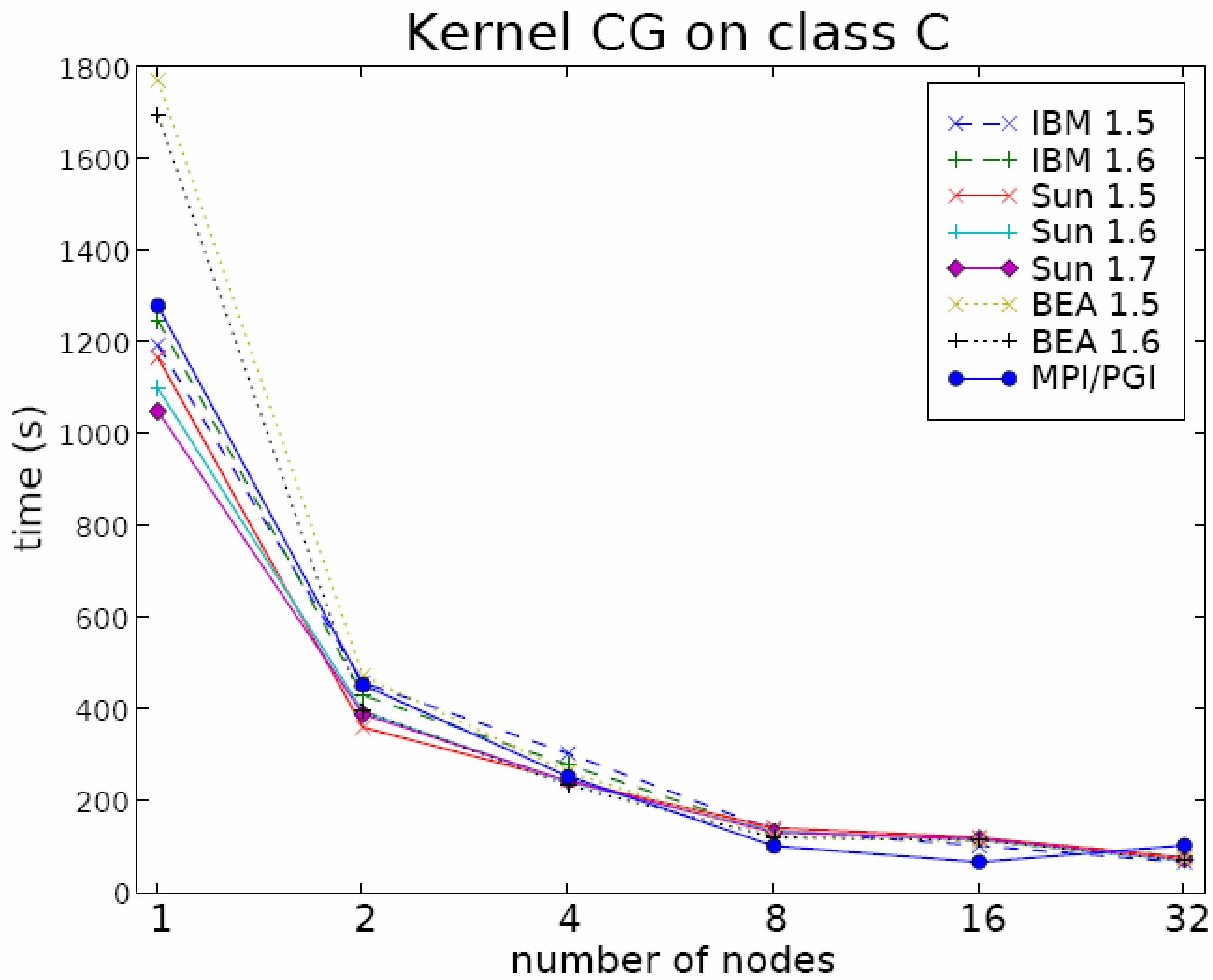


Data density distribution

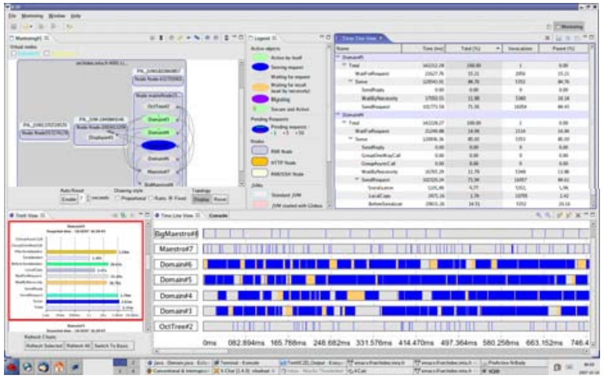


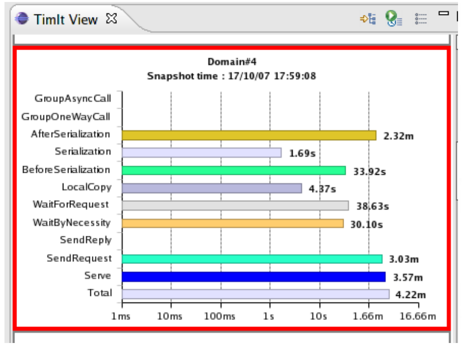
Message density distribution

Communication Intensive
CG Kernel (Conjugate Gradient)

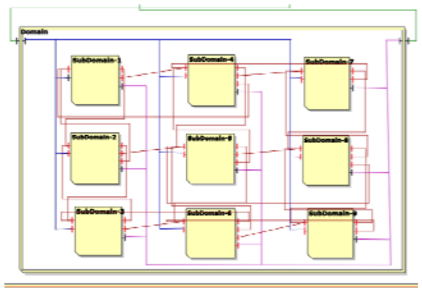


Summary-Perspective: Comprehensive Toolkit





Programming: Models & Tools



Parallel: Multi-Core & Distributed