Strong Programming Model to bridge Distributed & Multi-Core Computing

D. Caromel, et al.

Agenda

1. Background: OASIS, ActiveEon
2. Multi-Cores
3. Programming,
   Optimizing
   Scheduling
4. Enterprise Parallel Computing

Speed: Application + Development: Productivity

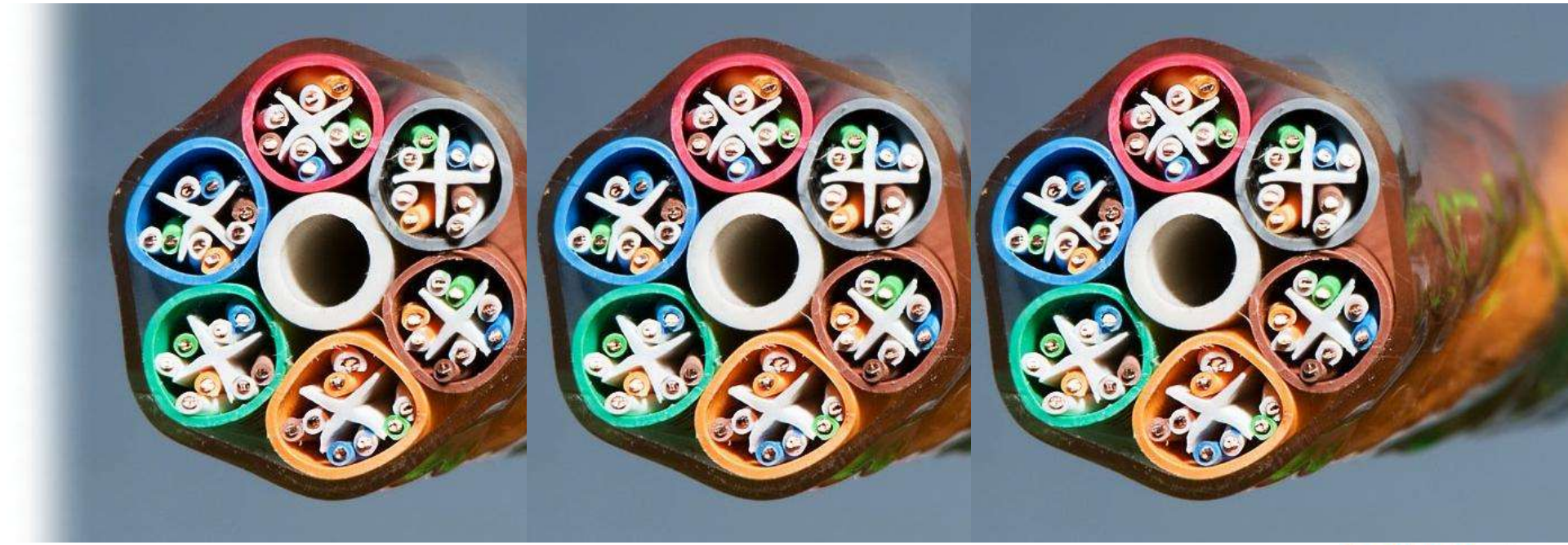
Key Objectives

- Parallel Programming Model and Tools
  - desesperatly needed
  - for the masses
  - for new architectures (Multi-cores)
- As Effective as possible:
  - Efficient
  - However Programmer Productivity is first KSF
- For both Multi-cores and Distributed
  - Actually the way around
- Some Handling of ``Large-scale'' (Grid, Clouds)

1. Background

OASIS Team & INRIA

- A joint team, Now about 35 persons
- 2004: First ProActive User Group
- 2009, April: ProActive 4.1, Distributed & Parallel: From Multi-cores to Enterprise GRIDs



OASIS Team Composition (35)

- Researchers (5):
  - D. Caromel (UNSA, Det. INRIA
  - E. Madelaine (INRIA)
  - F. Baude (UNSA)
  - F. Huet (UNSA)
  - L. Henrio (CNRS)
- PhDs (11):
  - Antonio Cansado (INRIA, Coni
  - Brian Amedro (SCS-Agos)
  - Cristian Ruz (INRIA, Conicyt)
  - Elton Mathias (INRIA-Cordi)
  - Imen Filali (SCS-Agos / FP7 S
  - Marcela Rivera (INRIA, Conicy
  - Muhammad Khan (STIC-Asia)
  - Paul Naoumenko (INRIA/Régio
  - Viet Dung Doan (FP6 Bionets)
  - Virginie Contes (SOA4ALL)
  - Guilherme Pezzi (AGOS, CIFR
- + Visitors + Interns

Located in Sophia Antipolis, between Nice and Cannes,
Visitors and Students Welcome!

ActiveEon
SCALE BEYOND LIMITS

Startup Company Born of INRIA

Activeeon
SCALE BEYOND LIMITS

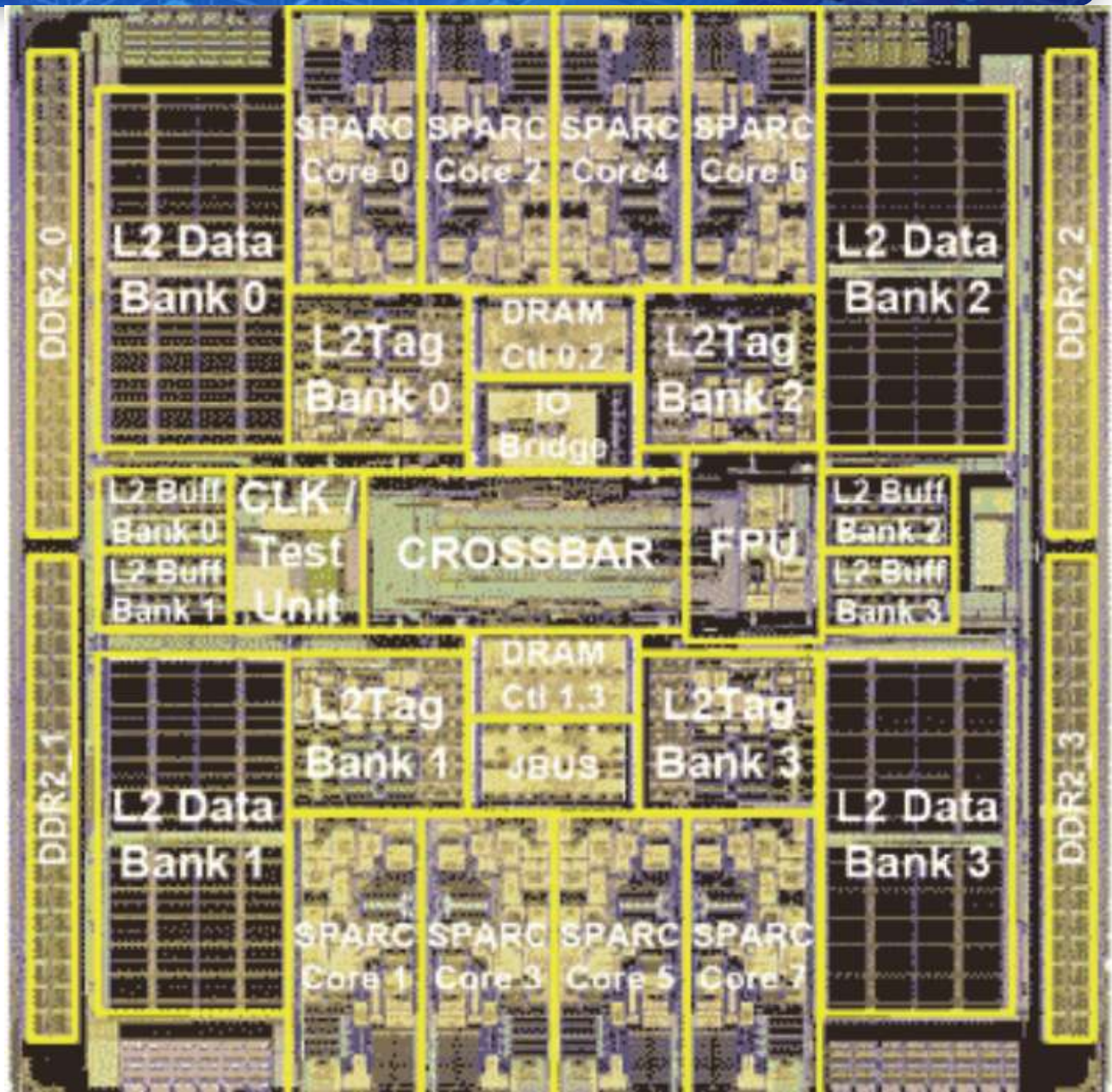
- Co-developing, Support for ProActive Parallel Suite
- Worldwide Customers: Fr, UK, Boston USA


Multi-Cores

Symetrical Multi-Core: 8-ways Niagara II

- 8 cores
- 4 Native threads per core

- Linux see 32 cores!



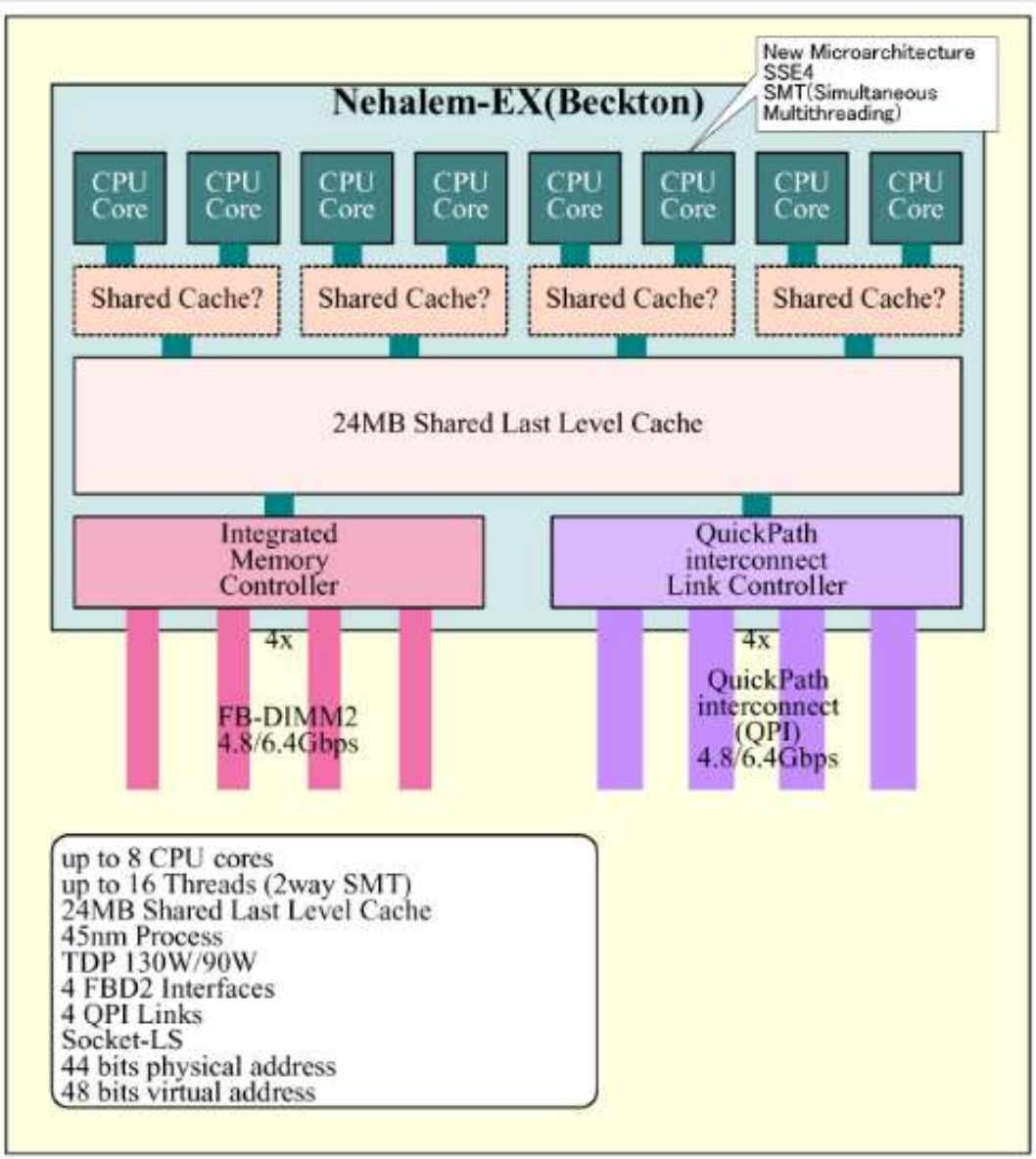
Activeeon
SCALE BEYOND LIMITS

Sun 16-core Rock: Fall 2009

- 16 cores
- 4 native threads per core
-
- ➔ 64 “Cores” or “Native Threads” at OS level

Intel 8-cores, 16-thread Nehalem-based Xeon processor confirmed (Feb. 2009)

- Highly
- NUMA
- Not an SMP:
- L1,
- L2, then
- L3 attached
- to a given
- core



Multi-Cores
A Few Key Points

- Moore's Law rephrased:
- Nb. of Cores double every 18 to 24 months
- Key expected Milestones: Cores per Chips (OTS)
  - 2010: 32 to 64
  - 2012: 64 to 128
  - 2014: 128 to 256
- 1 Million Cores Parallel Machines in 2012
- 100 M cores coming in 2020

- Multi-Cores are NUMA, and turning Heterogeneous (GPU)
- They are turning into SoC with NoC: NOT SMP!

ProActive Parallel Suite

Parallel Acceleration Toolkit in Java:

Parallelism:

Multi-Core+Distributed

Open Source Used in production by industry

Activeeon SCALE BEYOND LIMITS

ProActive Parallel Suite

OW2 Consortium

OW2: Object Web + Orient Ware

OW2 Consortium

Leading Open Source Middleware

Activeeon
SCALE BEYOND LIMITS

ProActive
Parallel Suite

OW2 Consortium

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 Contributors

Abhijeet Gaikwad (Option Pricing)
Abhishek-Rajeev Gupta
Antonio Cansado
Baptiste De Stefano
Bastien Sauvan
Brian Amedro (SPMD)
Cédric Dalmasso (Component)
Clement Mathieu (Core, GCM Deployment)
Elaine Isnard
Elton Mathias
Eric Madelaine
Etienne Vallette-De-Osia
Fabien Viale (Matlab, Scilab)
Fabrice Huet (Mobility, P2P)
Florin Bratu
Franca Perrina
Francoise Baude
Germain Sigety (Scheduling)
Guillaume Laurent
Guilherme Perretti Pezzi
Imen Filiali
Jean-Luc Scheefer (Scheduling)
Jean-Michael Guillamume
Johann Fradj (Scheduling)
Jonathan Martin
Julian Krzeminski
Kamran Qadir
Khan Muhammad
Laurent Vanni
Ludovic Henrio
Marcela Rivera
Mario Leyton (Skeleton)
Maxime Menant
Nicolas Dodelin
Olivier Helin
Paul Naoumenko
Regis Gascon
Tomasz Dobek
Vasile Jureschi (Technical Writer)
Viet Dong Doan
Vincent Cave (Legacy Wrapping)
Virginie Contes (OSGi, WS)
Yu Feng
Yulai Yuan
Zhihui Dai

Arnaud Contes (Core, Security, Debugging)
Christian Delbé (FT, Scheduling)
Emil Salageanu (IC2D, Agent)
Vladimir Bodnartchouk (IC2D, TimIt)

Alexandre di Costanzo (P2P, B&B)
Boutheina Bennour
Guillaume Chazarain (DGC)
Julien Vayssiere (MOP, Active Objects)

Lionel Mestre
Laurent Baduel (Group Communications)
Matthieu Morel (Initial Component Work)
Nadia Ranaldo (Core, Deployment)
Romain Quilici

ActiveEon SCALE BEYOND LIMITS
ProActive Parallel Suite
OW2 Consortium

2. Distributed and Parallel Active Objects

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.

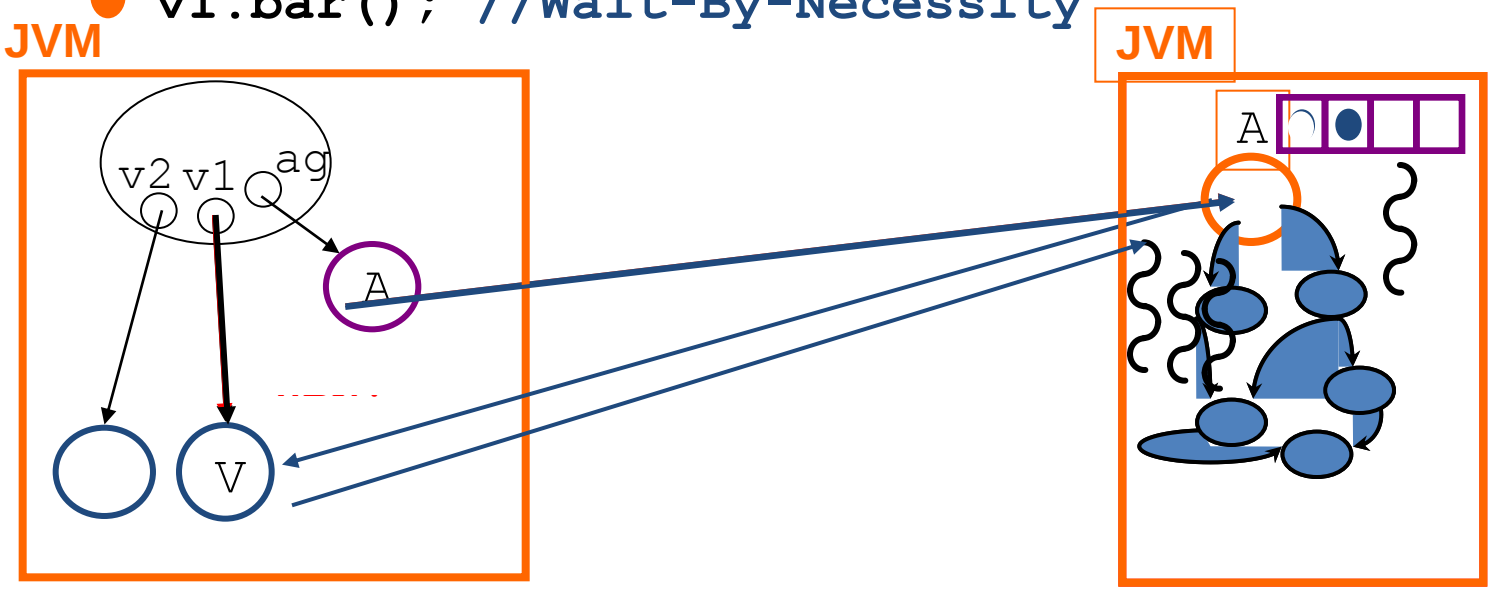
PROGRAMMING

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.

ProActive : Active objects

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



Java Object

Active Object

Req. Queue

Future Object

Proxy

Request

Thread

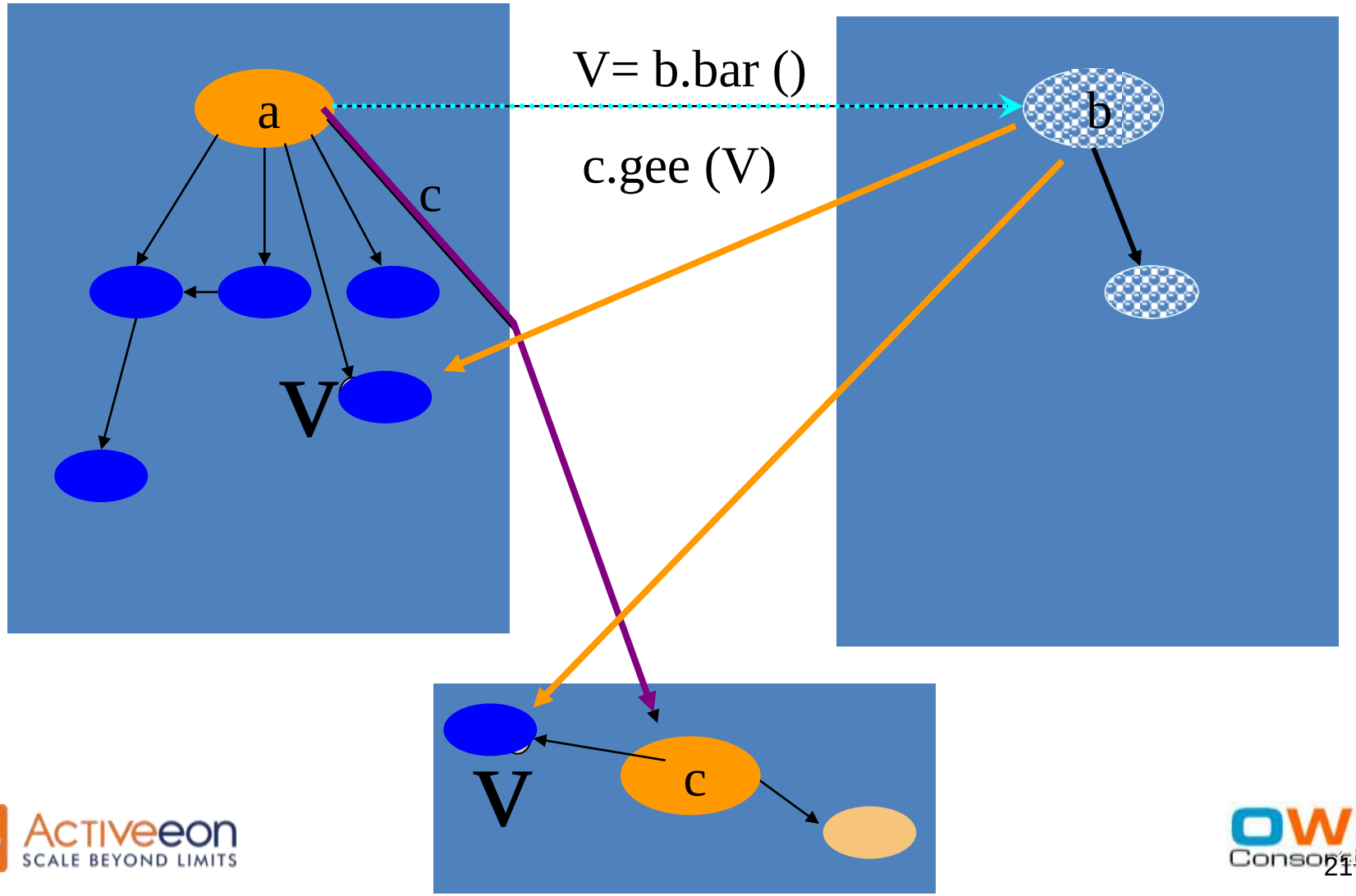
Wait-By-Necessity is a Dataflow Synchronization

19

First-Class Futures Update

Wait-By-Necessity: First Class Futures

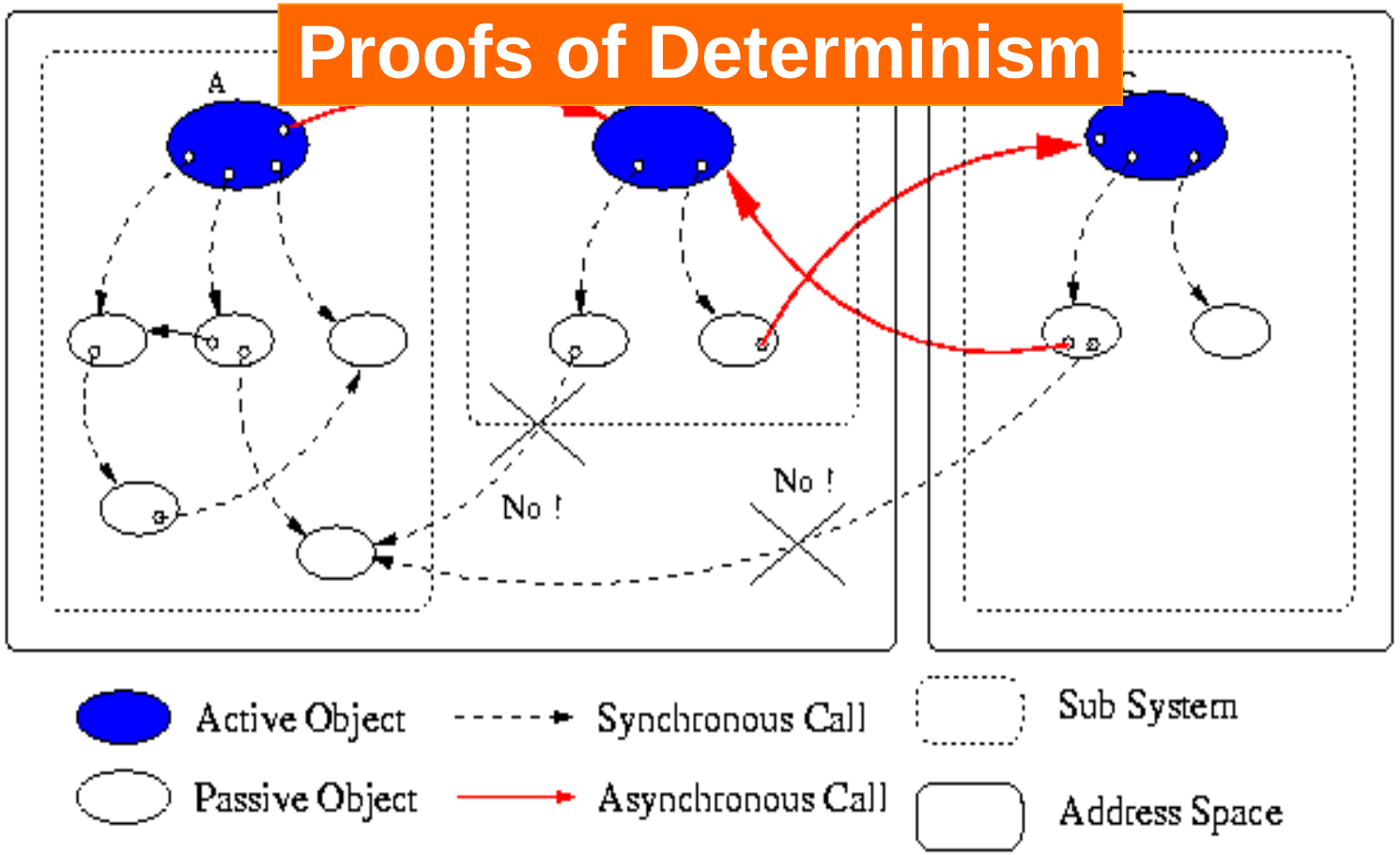
Futures are Global Single-Assignment Variables



Standard system at Runtime: No Sharing

NoC: Network On Chip

Proofs of Determinism



22

Key Point: Software Evolution

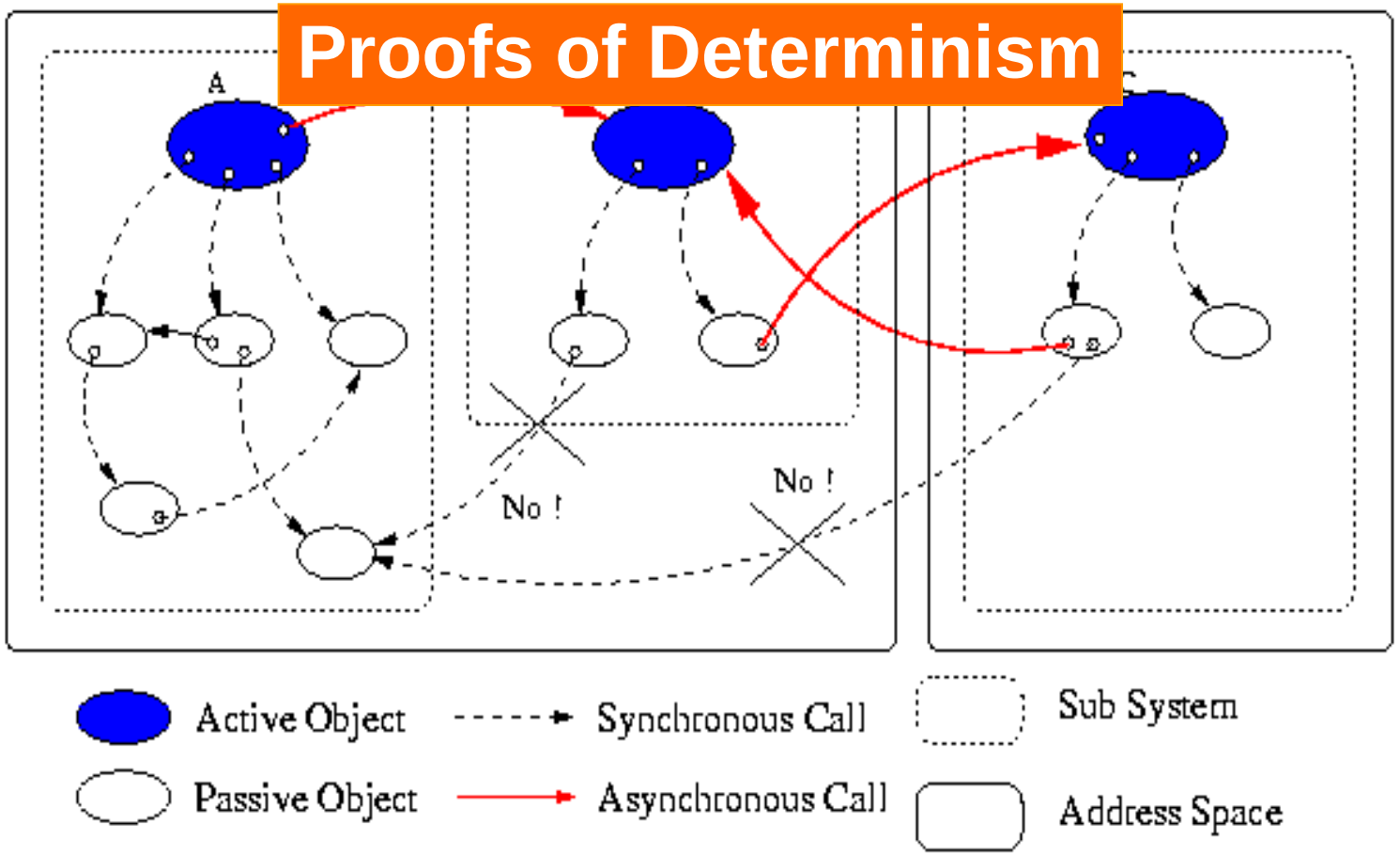
- Distributed To Multicores
- Multi-Cores: 32 (2010) to 64 to 128 to 256 (2014)
- Shift the execution from several multi-cores executing
- the same application simultaneously to a single, larger
- multi-core chip. An application requiring 128 cores to
- correctly execute, can be executed in 2012 on four 32 cores,
- and seamlessly executed in 2016 on a single 128-core chips

- ➔ Smooth evolutivity of applications
- Distributed and Multi-core Platforms
- ➔ Also for Data Scalability

Standard system at Runtime: No Sharing

NoC: Network On Chip

Proofs of Determinism



24

(2) ASP: Asynchronous Sequential Processes

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

Local

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

Creating an Activity

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

Sending a Request

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

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

Service

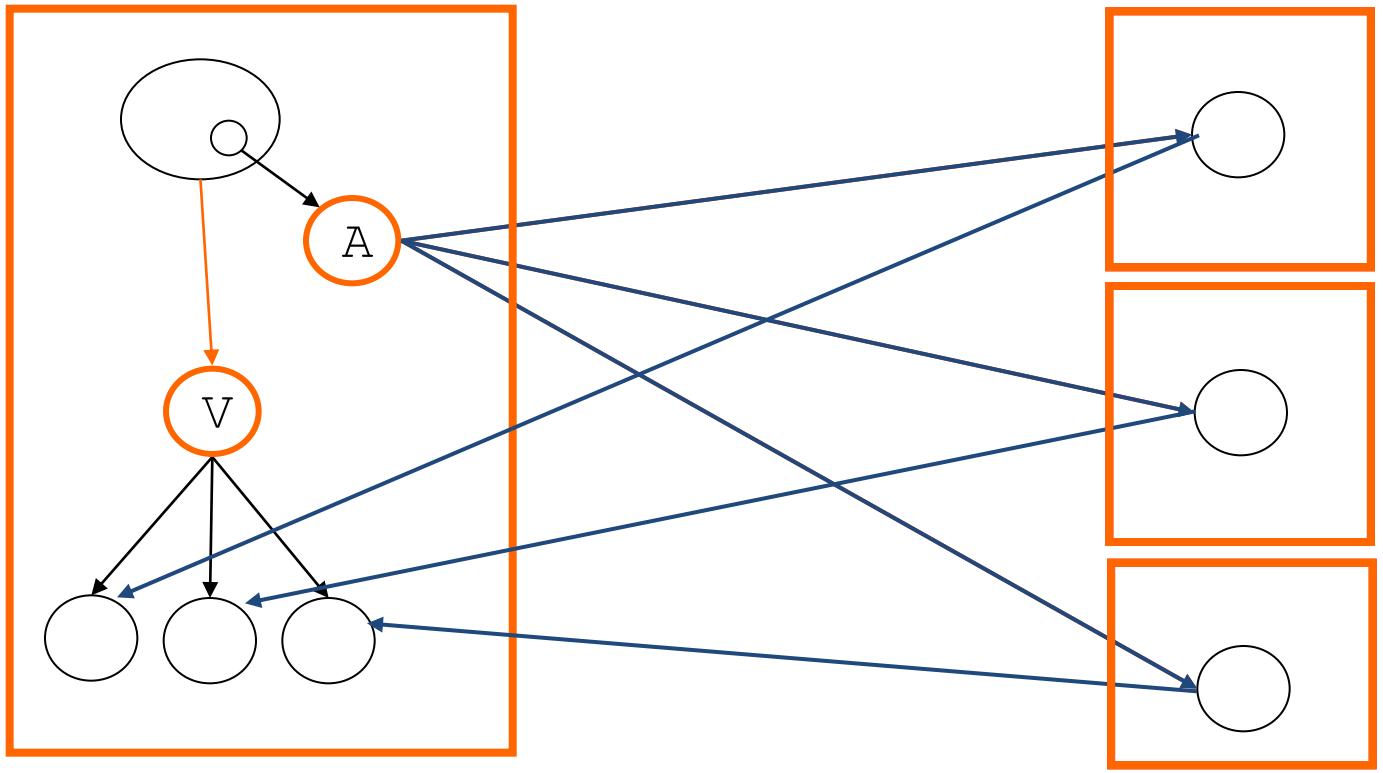
$$\frac{\sigma_\alpha(\iota) = fut(f_i^{\gamma\rightarrow\beta}) \qquad F_\beta(f_i^{\gamma\rightarrow\beta}) = \iota_f \qquad \sigma'_\alpha = Copy\&Merge(\sigma_\beta,\iota_f \ ; \ \sigma_\alpha,\iota)}{\begin{array}{l}\alpha[a_\alpha;\sigma_\alpha;\iota_\alpha;F_\alpha;R_\alpha;f_\alpha] \parallel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

Sending a Reply

TYPED ASYNCHRONOUS GROUPS

Creating AO and Groups

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



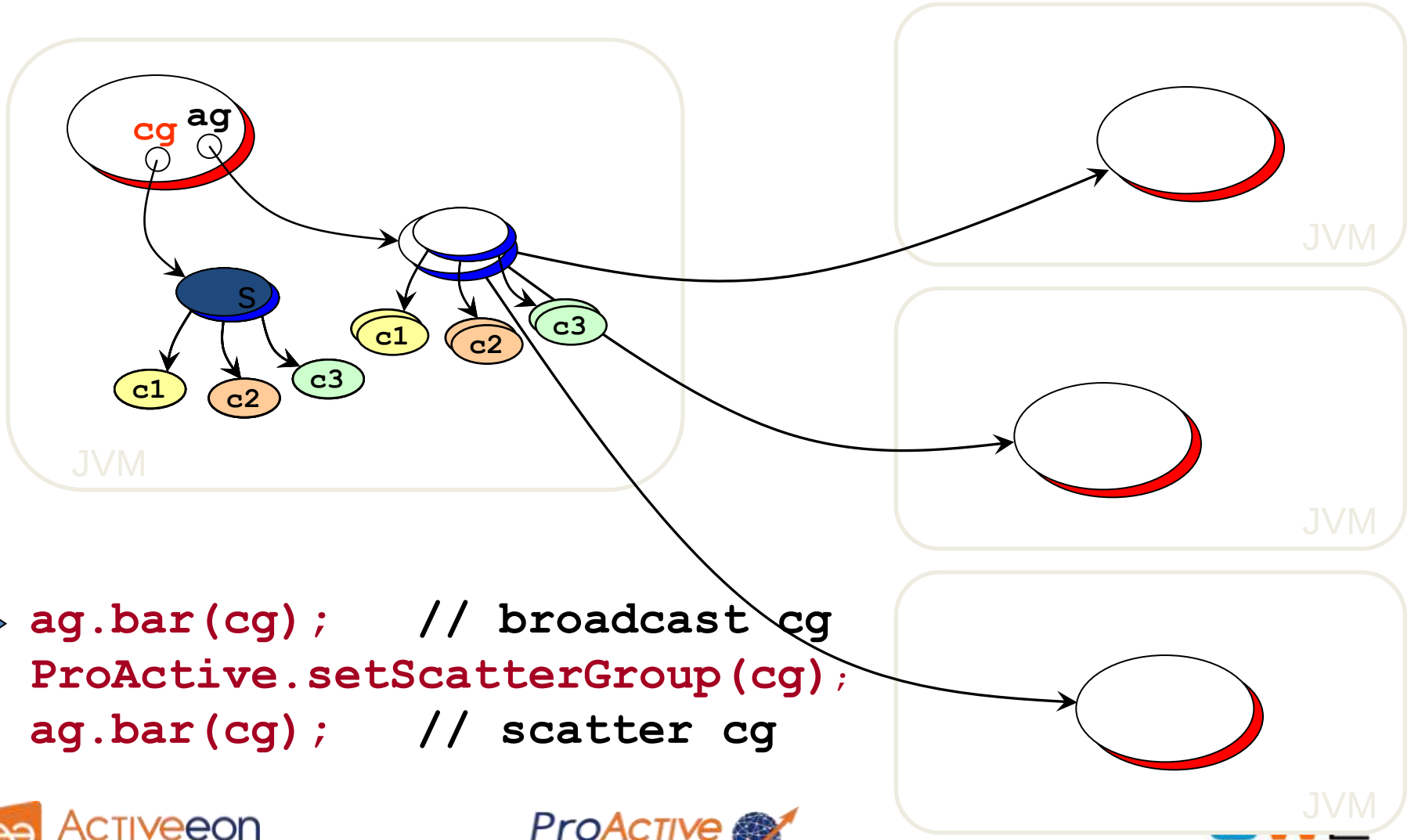
Typed Group Java or Active Object

Group, Type, and Asynchrony are crucial for Composition

27

Broadcast and Scatter

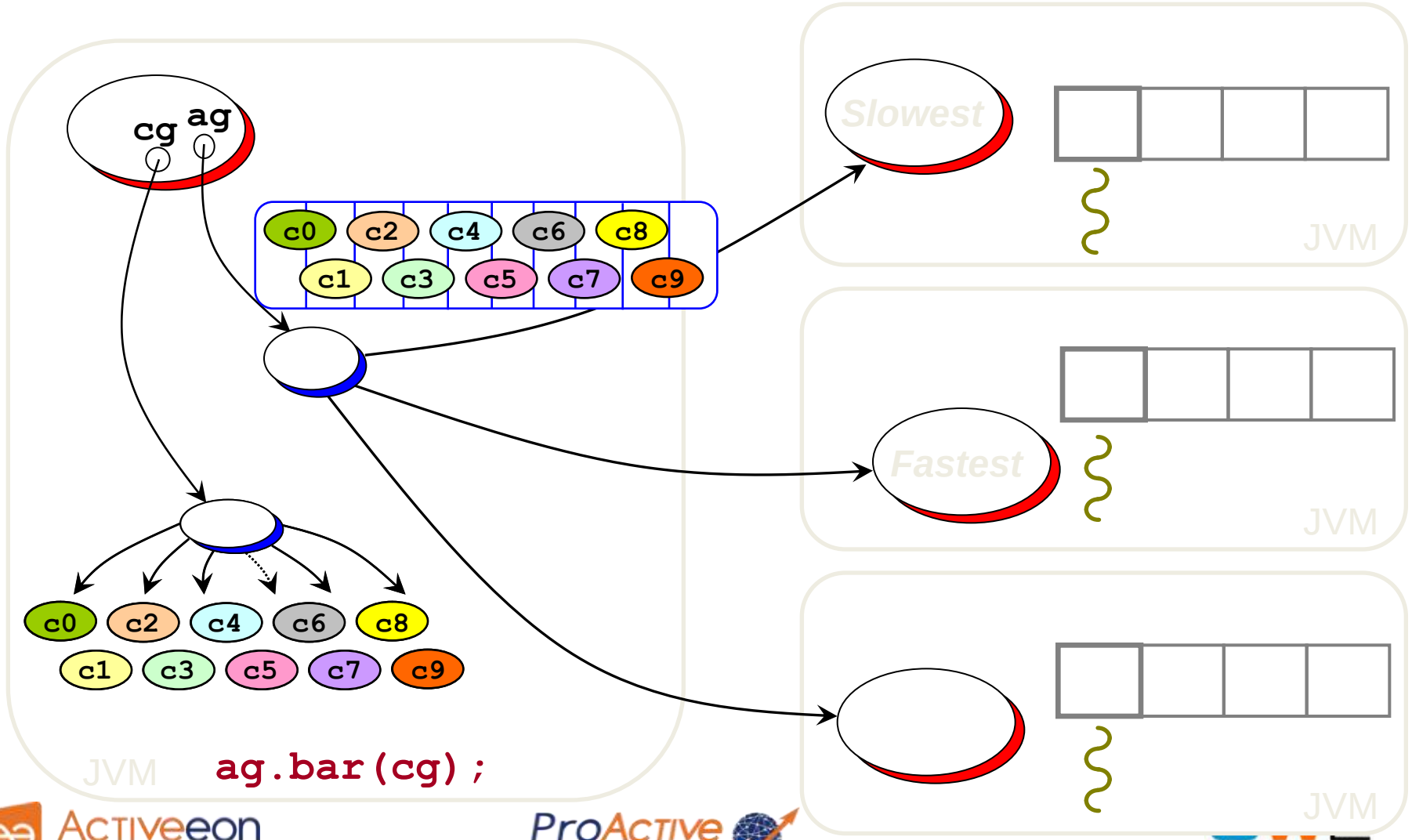
Broadcast is the default behavior
Use a group as parameter, Scattered depends on rankings




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


28

Dynamic Dispatch Group



29

Abstractions for Parallelism

The right Tool to do the Task right

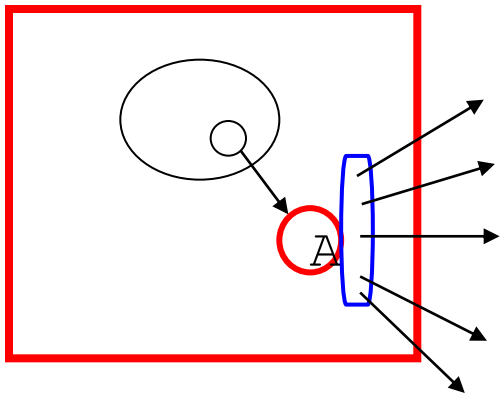
Object-Oriented SPMD

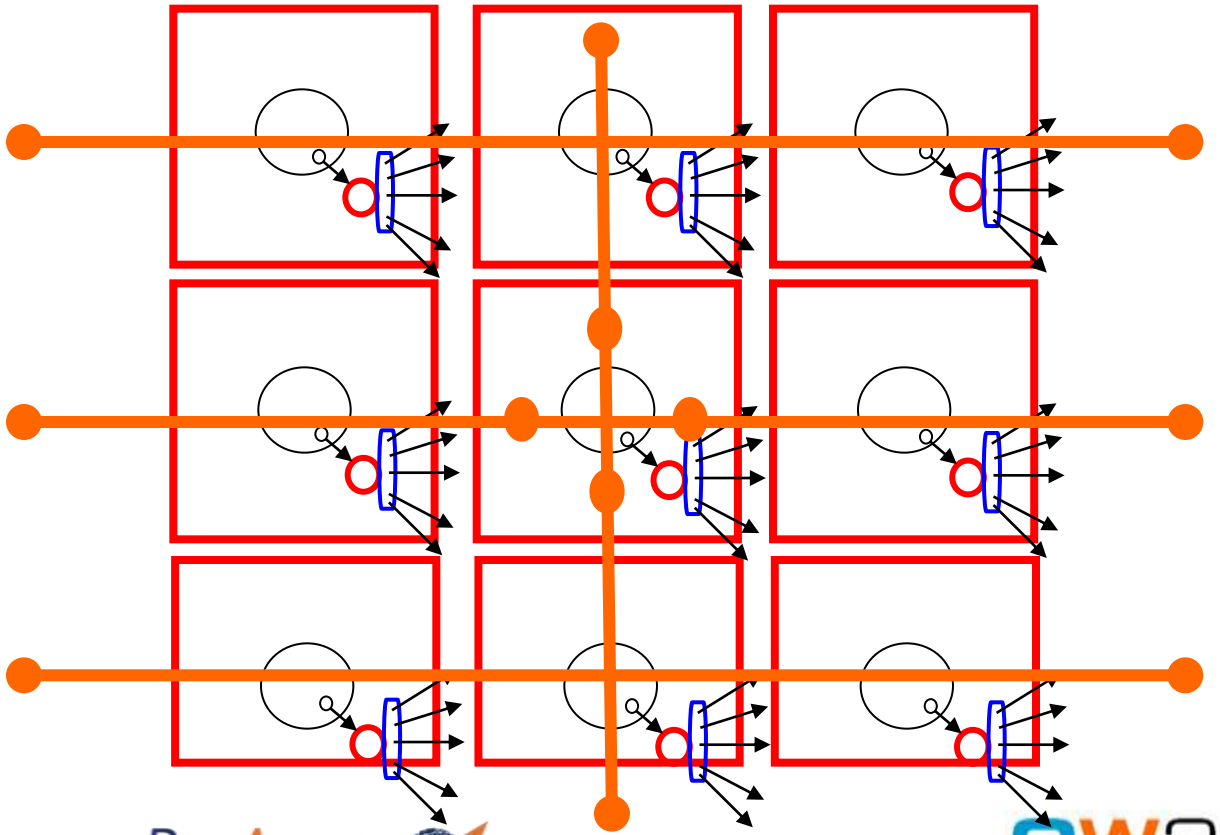
Key Point

- **“MPI and programming languages from the 60’s will not make it”**
- **Jack Dongarra,** 2/13/2009,
- Wake Forest University talk

OO SPMD: Object-Oriented 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

OO SPMD: Object-Oriented SPMD

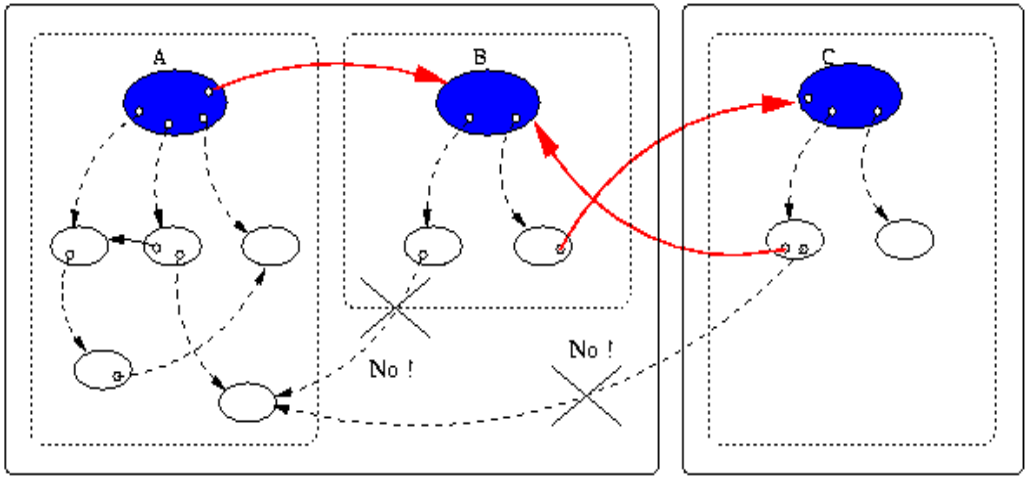
- Motivation
- **Use Enterprise technology (Java, Eclipse) for Numerical Parallel Computing**
- Able to express in Java MPI's Collective Communications:
  - **broadcast**
  - **scatter**
  - **gather**
  - **reduce**
  - **allscatter**
  - **allgather**
- Together with
- **Barriers, Topologies.**

34

Application Semantics rather than Low-Level Architecture-Based Optimization

- MPI: MPI_Send MPI_Recv MPI_Ssend MPI_Irecv
- MPI_Bsend MPI_Rsend MPI_Isend MPI_Ibsend
- What we propose:
  - High-level Information from Application Programmer
  - Tower Self-Adapting parallel applications

- Examples:
  - ro.foo (***ForgetOnSend*** (params));
  - ***ActiveObject.exchan***



➔ Optimizations for Both
➔ Distributed &
➔ Multi-Core

ActiveEon SCALE BEYOND LIMITS
ProActive Parallel Suite
OW2 Consortium

Key Point:
Infrastructure Independence

- Application Abstractions
  - I give you this data and I no longer need it
- Not Infrastructure Abstractions
  - I asynchronous send you this and I do not lock the buffer

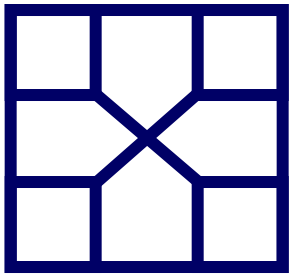
NAS Parallel Benchmarks

- Experimented on 3D ElectroMagnetism, and Nasa Benchmarks

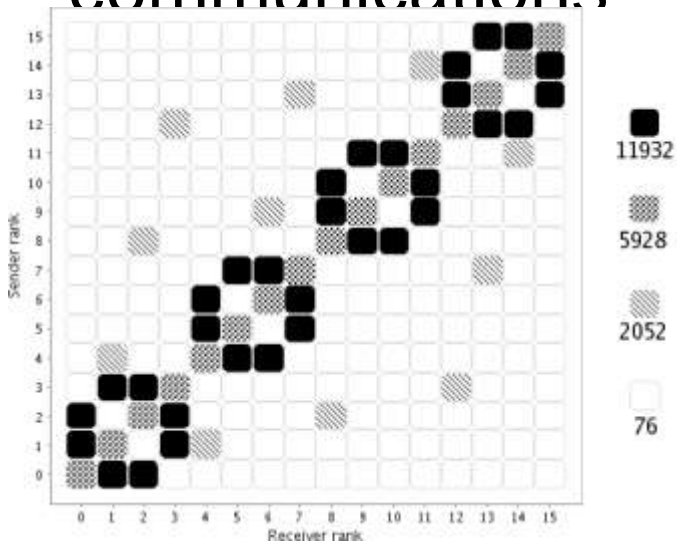
- **Designed by NASA to evaluate benefits of high performance systems**
- **Strongly based on CFD**
- **5 benchmarks (kernels) to test different aspects of a system**
- **2 categories or focus variations:**
  - **communication intensive and computation intensive**

Communication Intensive
CG Kernel (Conjugate Gradient)

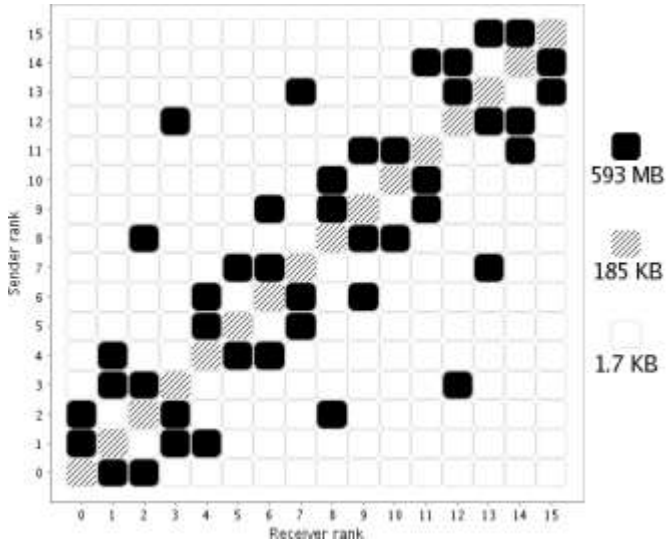
- Floating point operations
- Eigen value computation
- High number of unstructured communications



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

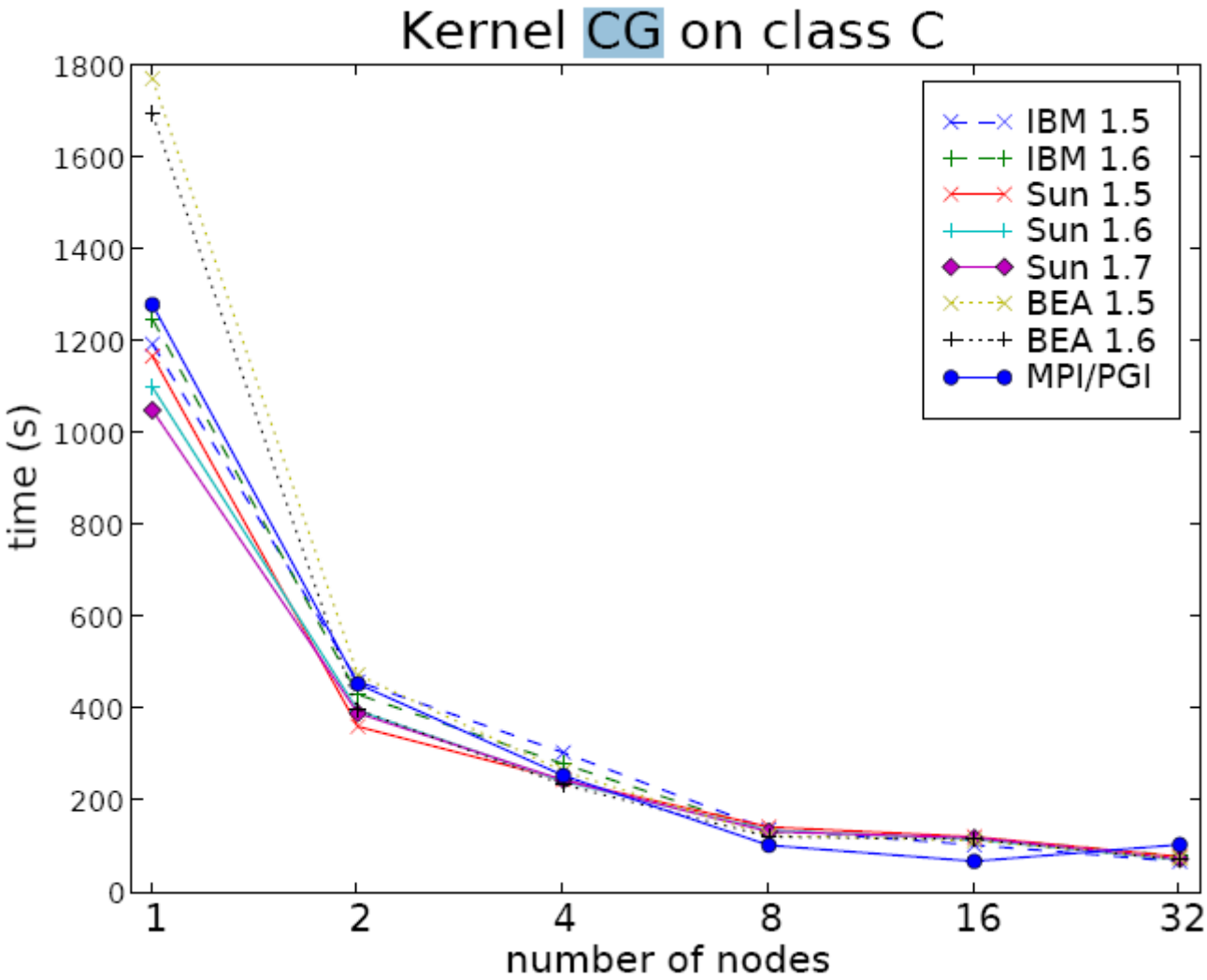


Message density distribution



Data density distribution

Communication Intensive
CG Kernel (Conjugate Gradient)



Comparable Performances

Key Point: Locality will more than ever be Fundamental

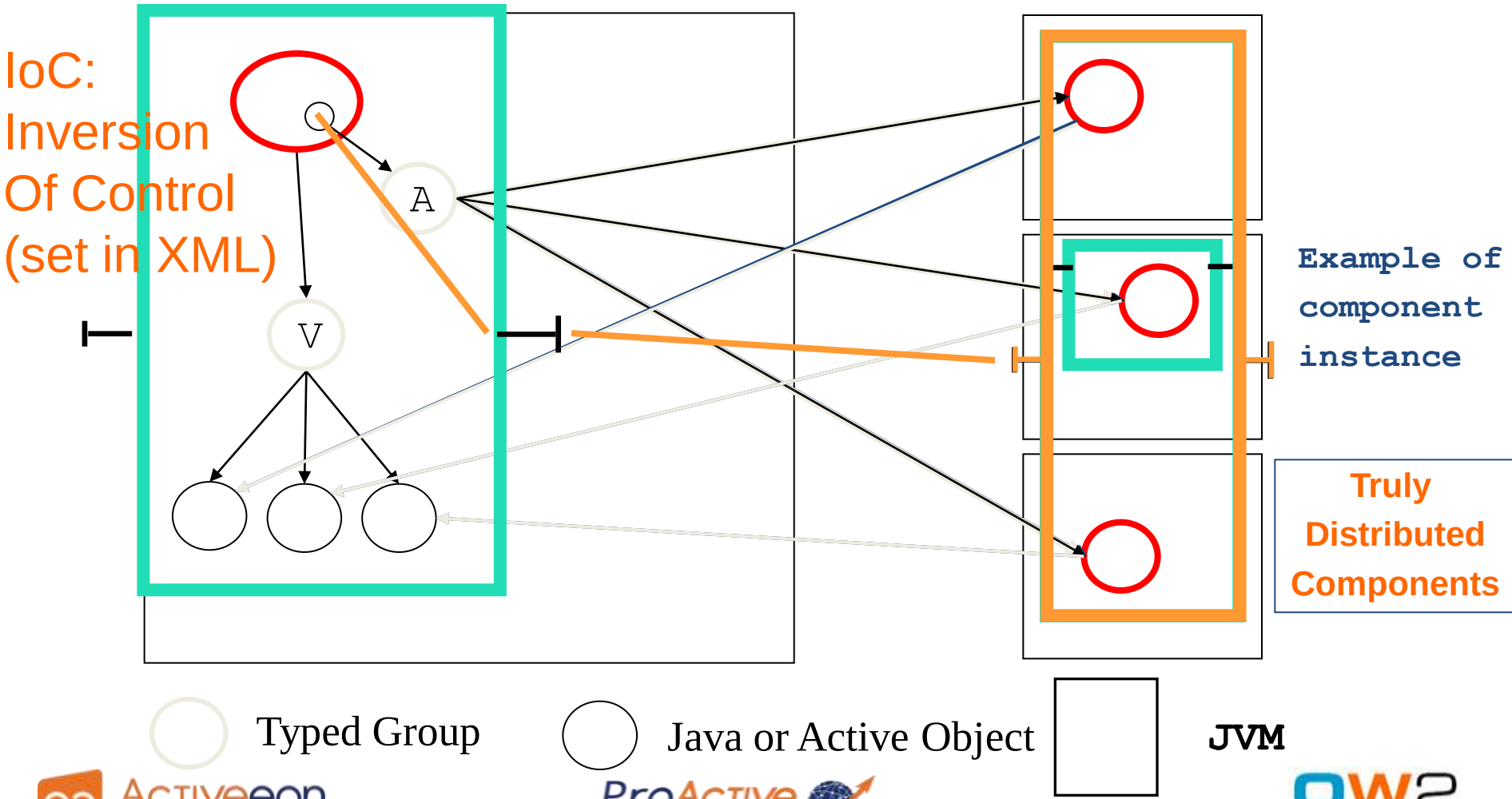
- Let the programmer control it
- No global shared memory
- **PGAS like**
- **Partitioned Global Address Space**
- But with more
- Flexibility, Dynamicity and Control

- One can envision: Spatial view of multicore

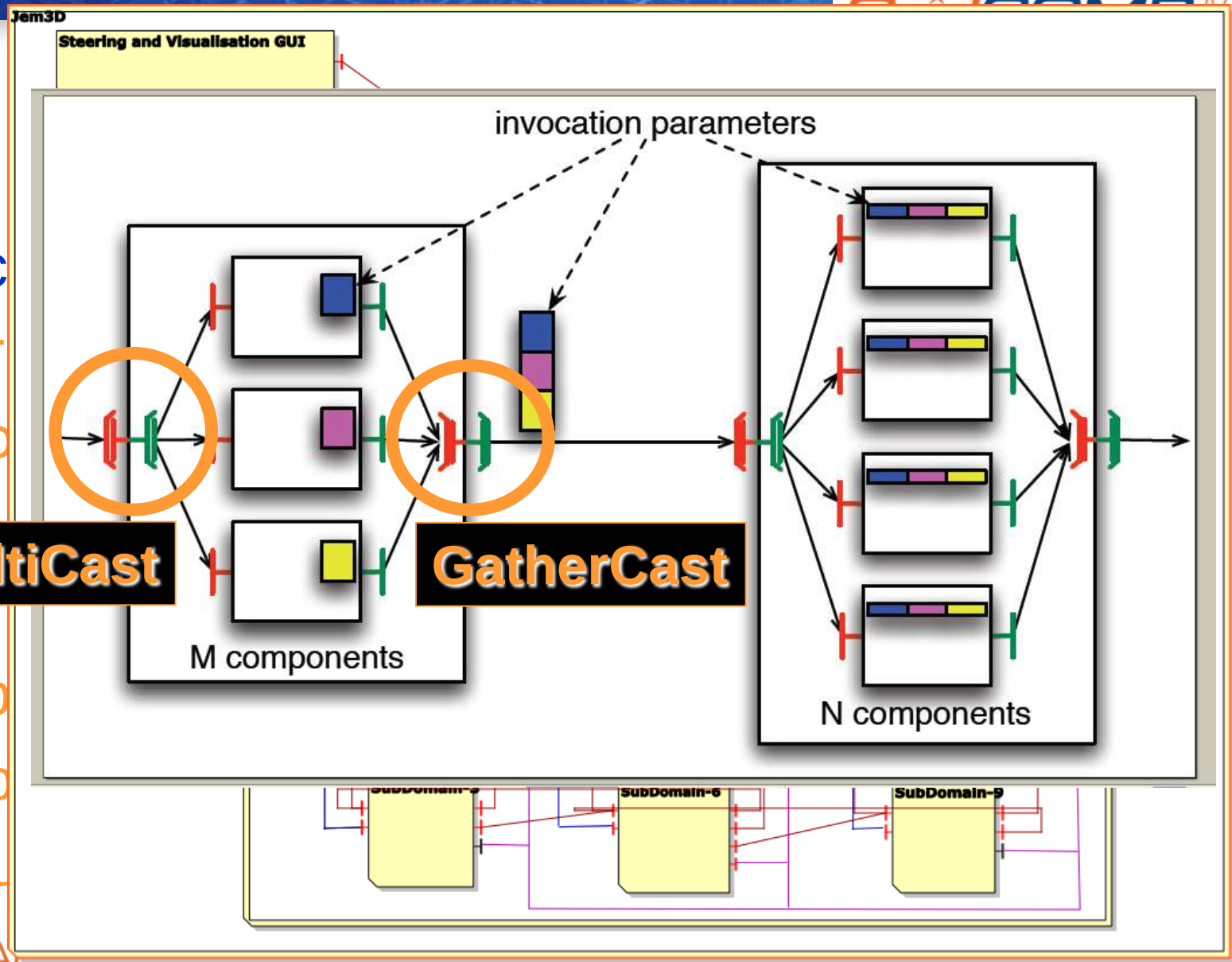
❑Research for
❑High-Level Parallel Abstractions

GridCOMP Partners

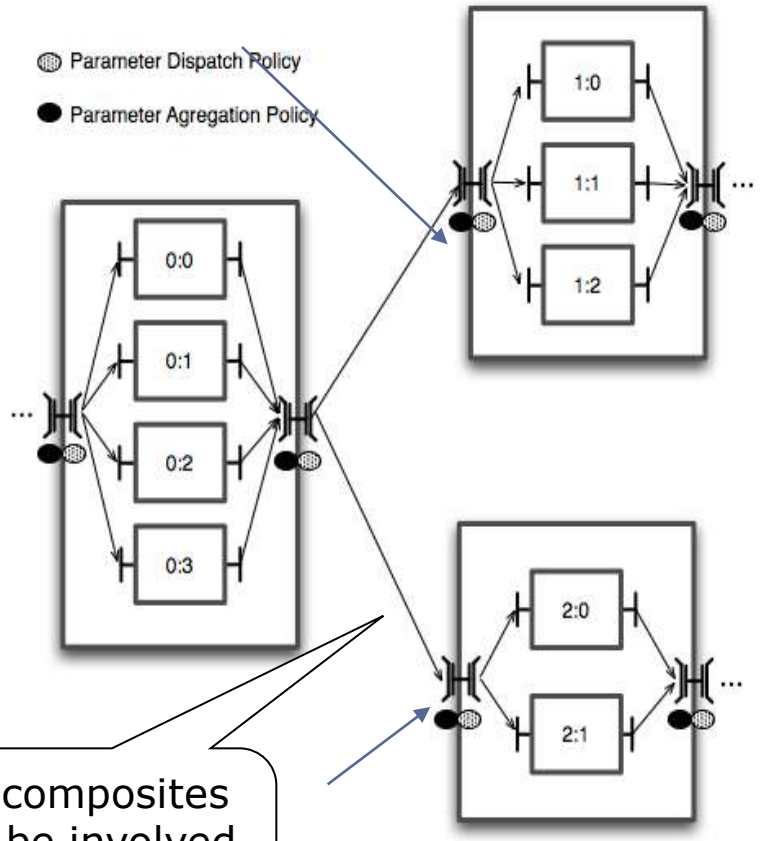
Objects to Distributed Components

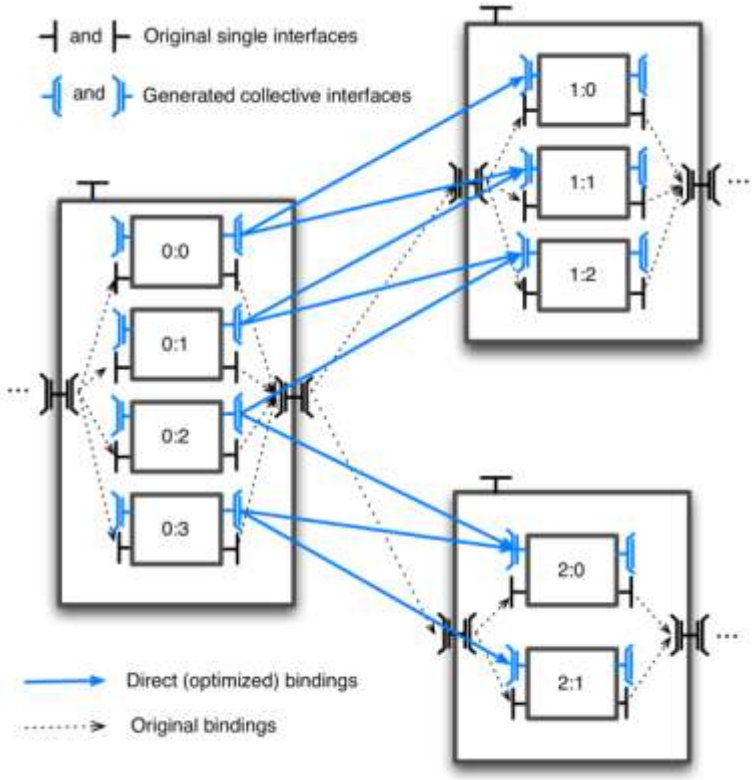


43


Jem3D
Steering and Visualisation GUI
invocation parameters
MultiCast
GatherCast
M components
N components
SubDomain-6
SubDomain-9

Optimizing MxN Operations





2+ composites can be involved in the Gather-multicast

Key Points about Parallel Components

- Parallelism is *captured* at the Module *interface*
- Identical to Typing for functional aspects
- *Composition*, in a parallel word, becomes possible
- *Configuration* of the Parallel *aspect*


Optimizing

ProActive Parallel Suite

PROGRAMMING

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

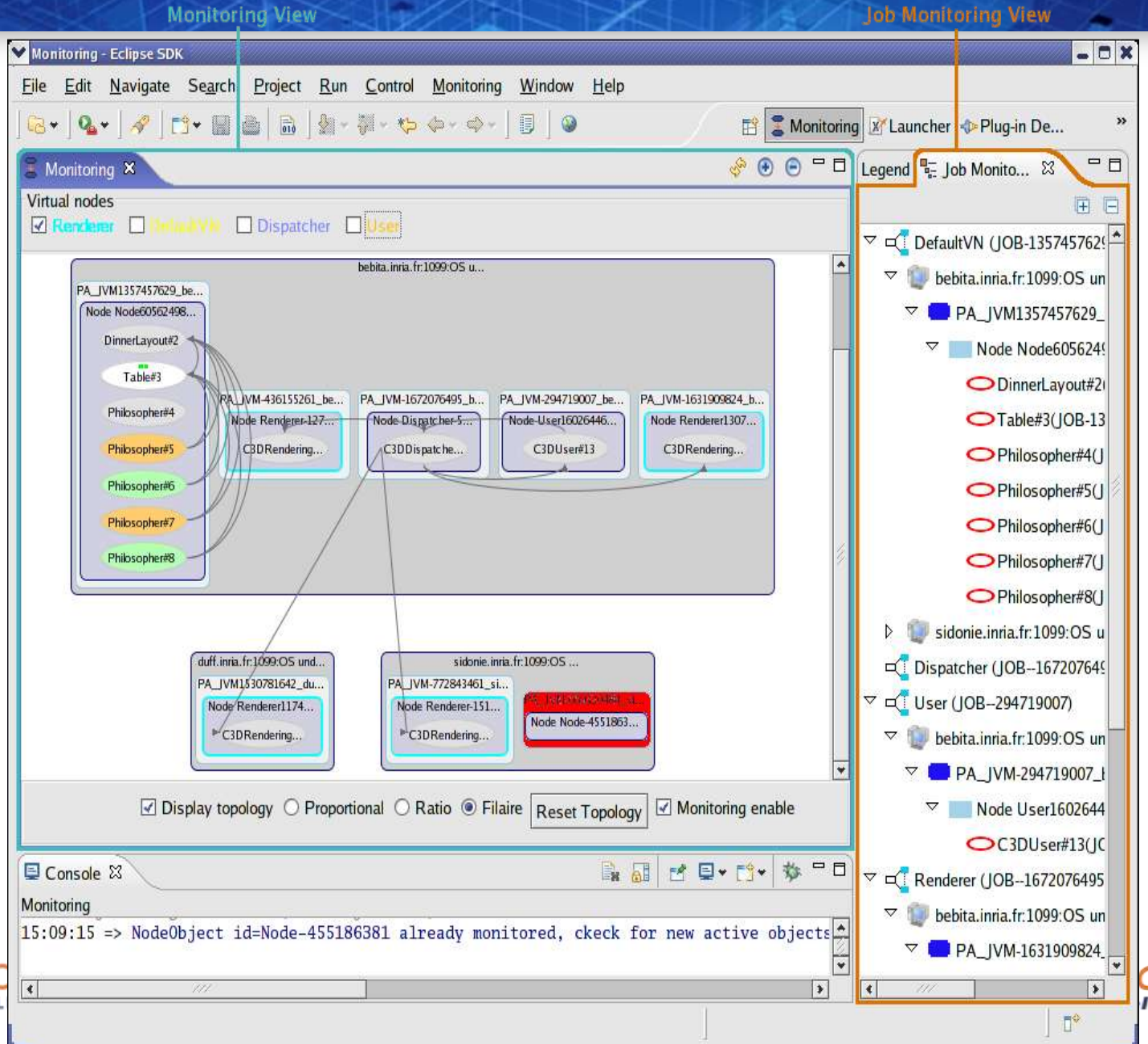
Featuring: Async. comms, Master-Worker, Monte-Carlo, SPMD, components and legacy code wrapping.

OPTIMIZING

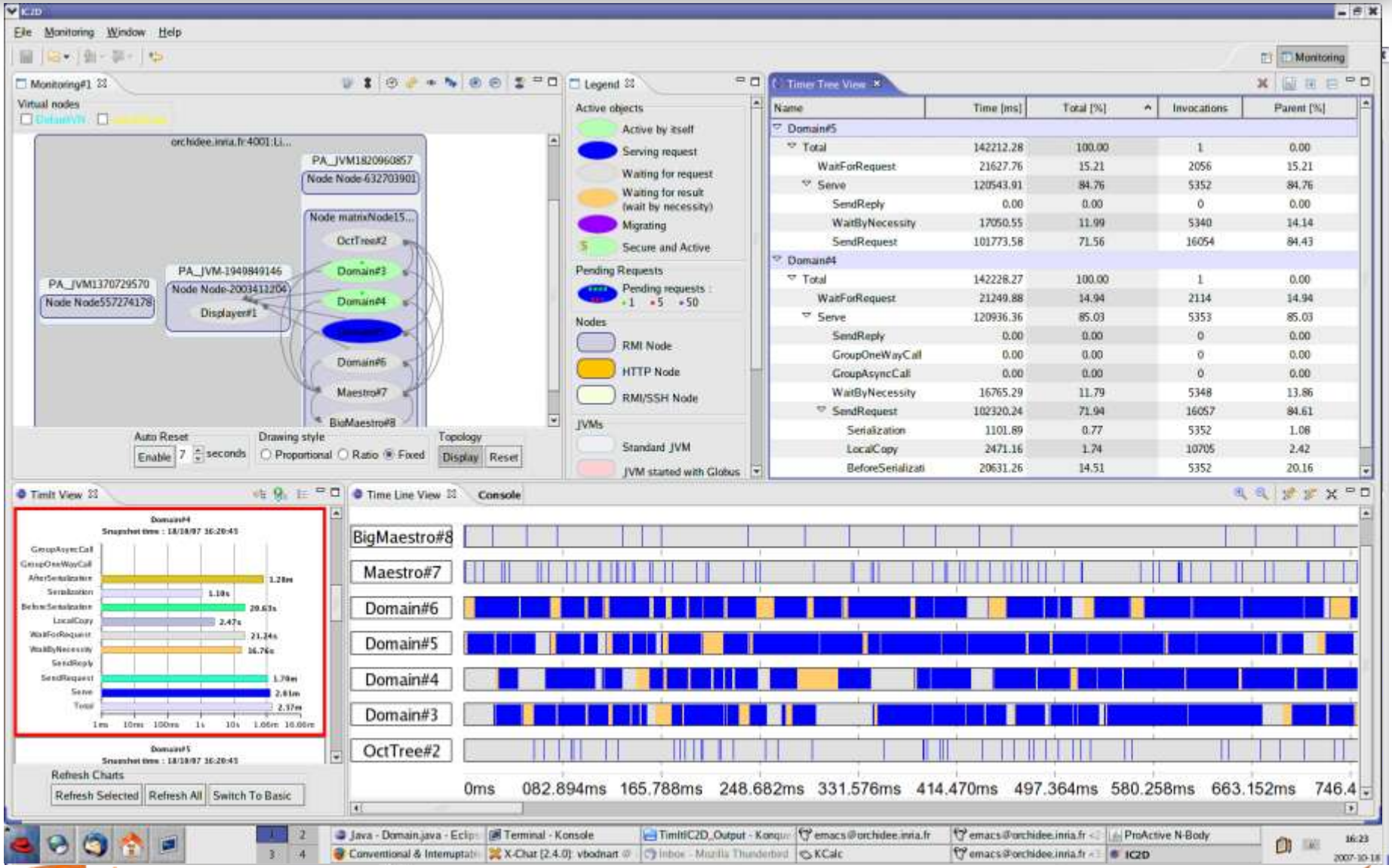
Eclipse GUI (IC2D)
for Developing, Debugging, Optimizing your parallel applications.

Featuring: graphical monitoring and benchmarking with report generation.

IC2D


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
Dispatcher
User
bebita.inria.fr:1099:OS u...
PA_JVM1357457629_be...
Node Node60562498...
DinnerLayout#2
Table#3
Philosopher#4
Philosopher#5
Philosopher#6
Philosopher#7
Philosopher#8
PA_JVM-436155261_be...
Node Renderer-127...
C3DRendering...
PA_JVM-1672076495_b...
Node Dispatcher-5...
C3DDispatche...
PA_JVM-294719007_be...
Node User16026446...
C3DUser#13
PA_JVM-1631909824_b...
Node Renderer1307...
C3DRendering...
duff.inria.fr:1099:OS und...
PA_JVM1530781642_du...
Node Renderer1174...
C3DRendering...
sidonie.inria.fr:1099:OS ...
PA_JVM-772843461_si...
Node Renderer-151...
C3DRendering...
Node Node-4551863...
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-1357457629
bebita.inria.fr:1099:OS un
PA_JVM1357457629_
Node Node605624
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--167207649
User (JOB--294719007)
bebita.inria.fr:1099:OS un
PA_JVM-294719007_
Node User1602644
C3DUser#13(JC
Renderer (JOB--1672076495
bebita.inria.fr:1099:OS un
PA_JVM-1631909824_

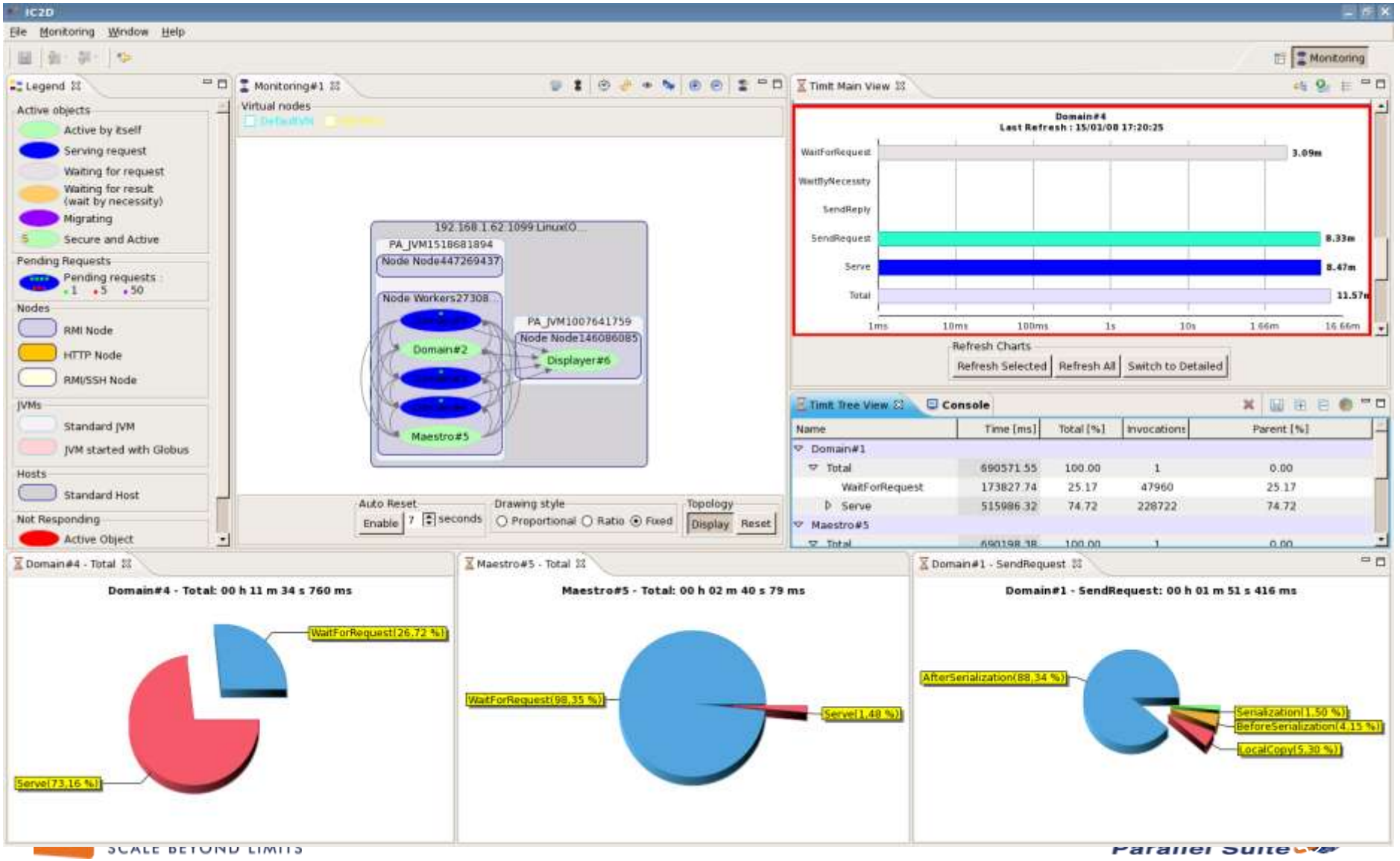
IC2D


IC2D
File Monitoring Window Help
Monitoring
Monitoring#1
Virtual nodes
orchidee.inria.fr:4001:Li...
PA_JVM1820960857
Node Node-632703901
Node matrixNode15...
OctTree#2
Domain#3
Domain#4
Domain#6
Maestro#7
BioMaestro#8
PA_JVM-1940849146
Node Node-2003411204
Displayer#1
PA_JVM1370729570
Node Node557274178
Auto Reset
Enable 7 seconds
Drawing style
Proportional Ratio Fixed
Topology
Display Reset
Legend
Active objects
Active by itself
Serving request
Waiting for request
Waiting for result (wait by necessity)
Migrating
Secure and Active
Pending Requests
Pending requests : 1 5 50
Nodes
RMI Node
HTTP Node
RMI/SSH Node
JVMs
Standard JVM
JVM started with Globus
Timer Tree View
Name | Time [ms] | Total [%] | Invocations | Parent [%]
Domain#5
Total 142212.28 100.00 1 0.00
WaitForRequest 21627.76 15.21 2056 15.21
Serve 120543.91 84.76 5352 84.76
SendReply 0.00 0.00 0 0.00
WaitByNecessity 17050.55 11.99 5340 14.14
SendRequest 101773.58 71.56 16054 84.43
Domain#4
Total 142228.27 100.00 1 0.00
WaitForRequest 21249.88 14.94 2114 14.94
Serve 120936.36 85.03 5353 85.03
SendReply 0.00 0.00 0 0.00
GroupOneWayCall 0.00 0.00 0 0.00
GroupAsyncCall 0.00 0.00 0 0.00
WaitByNecessity 16765.29 11.79 5348 13.86
SendRequest 102320.24 71.94 16057 84.61
Serialization 1101.89 0.77 5352 1.08
LocalCopy 2471.16 1.74 10705 2.42
BeforeSerializati 20631.26 14.51 5352 20.16
TimIt View
Time Line View
Console
BigMaestro#8
Maestro#7
Domain#6
Domain#5
Domain#4
Domain#3
OctTree#2
0ms 082.894ms 165.788ms 248.682ms 331.576ms 414.470ms 497.364ms 580.258ms 663.152ms 746.4
Refresh Charts
Refresh Selected Refresh All Switch To Basic
Java - Domain.java - Eclip
Terminal - Konsole
TimItIC2D_Output - Konqu
emacs@orchidee.inria.fr
emacs@orchidee.inria.fr
ProActive N-Body
Conventional & Interruptab
X-Chat [2.4.0]: vbodnart
Inbox - Mozilla Thunderbird
KCalc
emacs@orchidee.inria.fr
IC2D
16:23
2007-10-18

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
2
4
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
6
2
4
5
AllBodiesCount
InternalBodiesCount
UserBodiesCount
HalfBodiesCount

Pies for Analysis and Optimization



Video 1:
IC2D Optimizing
Monitoring, Debugging, Optimizing

ACTIVEEON
SCALE BEYOND LIMITS

ProActive
Parallel Suite

OW2
Consortium

3. Scheduling

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.

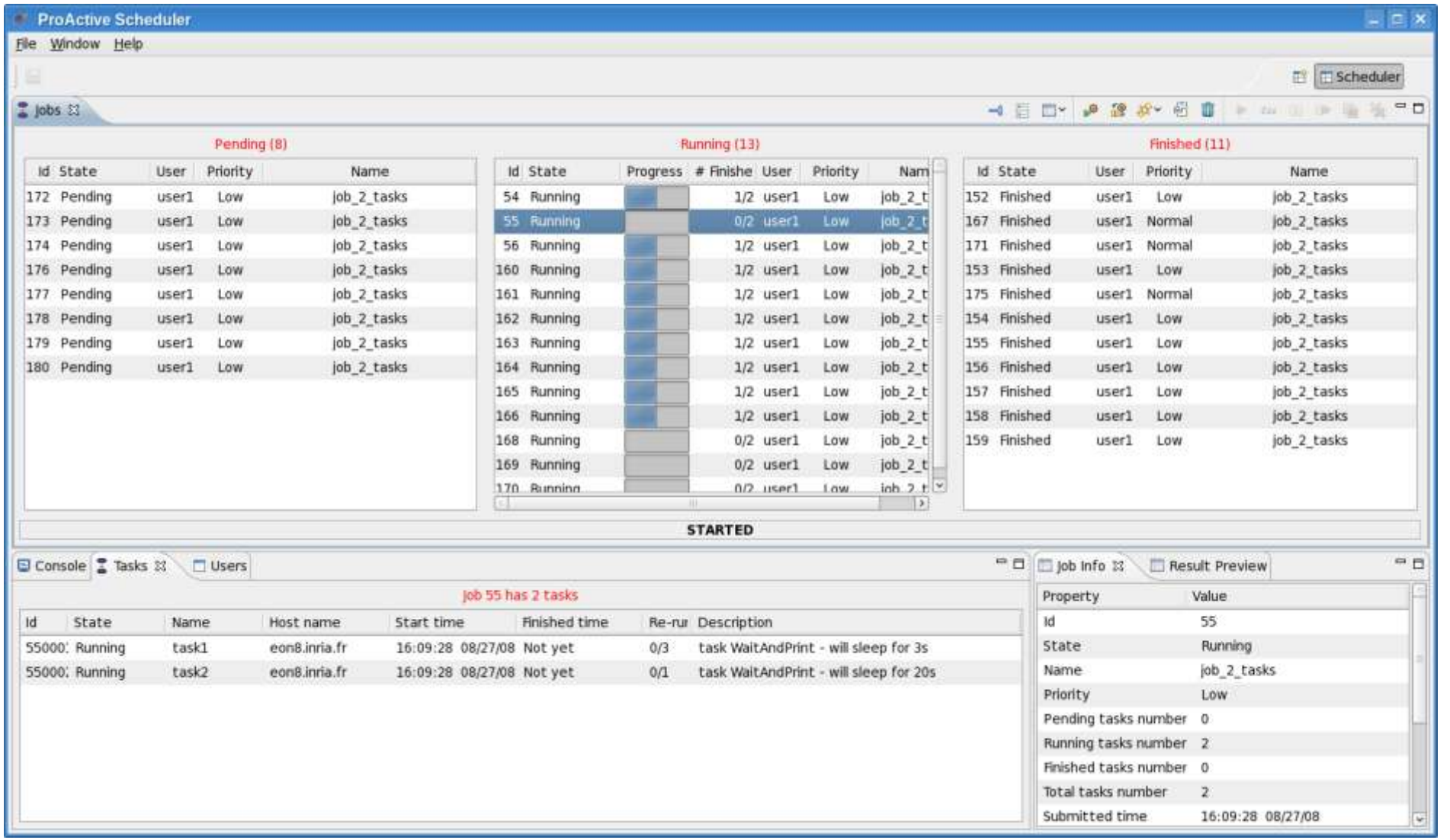
Featuring: graphical monitoring and benchmarking with report generation.

SCHEDULING

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

Featuring: graphical user interface, resource acquisition and virtualization.

Scheduler: User Interface



ActiveEon SCALE BEYOND LIMITS

ProActive Parallel Suite

OW2 Consortium

Video 2:
Scheduler, Resource Manager

4. Enterprise Grids, Clouds: Standards & Amazon EC2

GCM Standardization

Grid Component Model

Overall, the standardization is supported by industrials:

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

PLUGTESTS
INTEROP EVENTS

GRIDS for Finance
& Telecommunications

V GRI

20-24 Oc

INRIA – Sophia

Information & registration : www.etsi.org/plugtests/GRID2008/GRID.htm

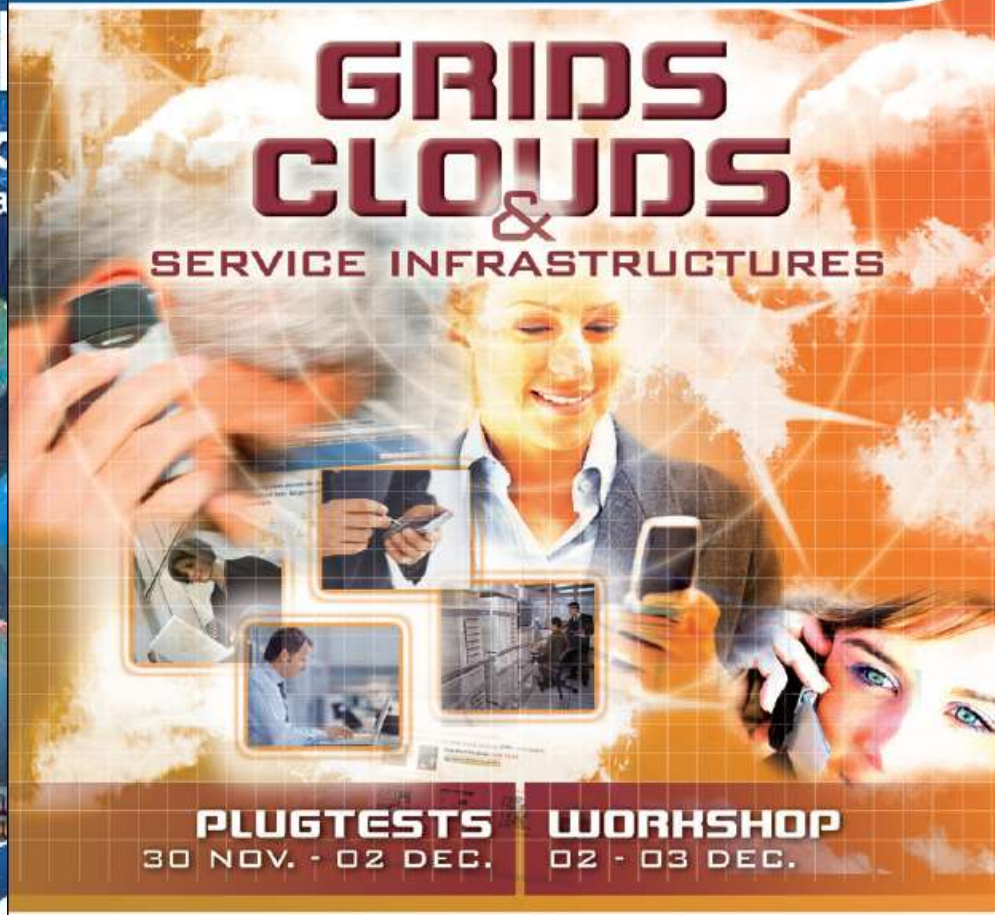
With the support of

European partners

Industrial sponsors

PLUGTESTS
INTEROP EVENTS

GRIDS CLOUDS & SERVICE INFRASTRUCTURES



PLUGTESTS
30 NOV. - 02 DEC.

WORKSHOP
02 - 03 DEC.

Information & registration at http://www.etsi.org/plugtests/GRID09/GRID.htm

30 NOV. - 03 DEC. 2009
ETSI - FRANCE

ETSI
World Class Standards

Summary

Summary of Key Points

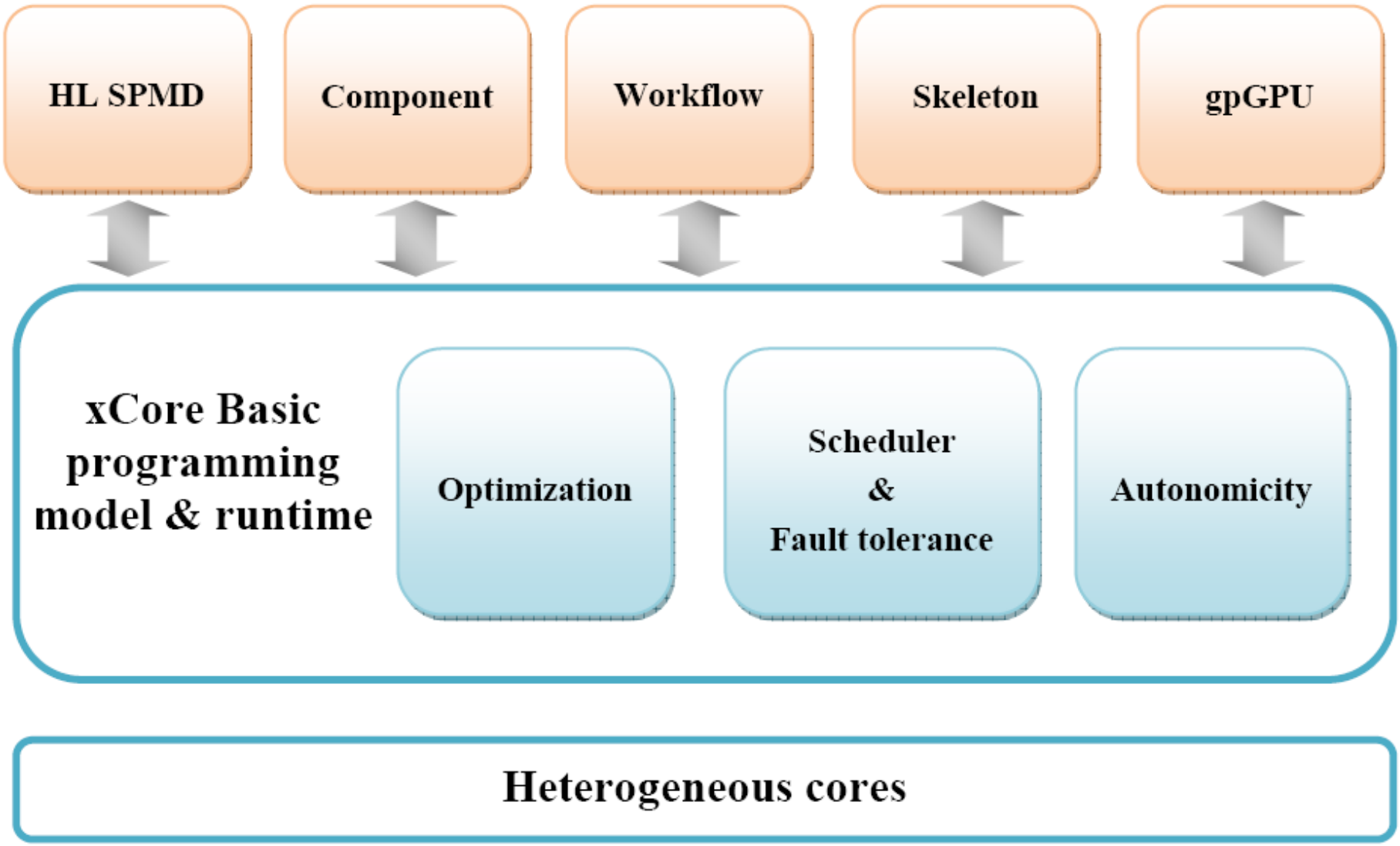
- Multi-Cores are NUMA, and turning Heterogeneous (GPU)
  - They are turning into SoC with NoC: NOT SMP!
  - Smooth evolution needed: Distributed to Multi-core

- A need for a unified Parallel Abstraction:
- Multi-Core + Distributed
- Shall MPI and OpenMP RIP
  - Application Abstractions Not Infrastructure Abstractions
  - Maintain strong Programmer control on Locality

Other Evolutions

- **Scheduling of Asynchronous Tasks, Workflows, Dynamic Data Driven Execution**
- **Fault-Tolerance + Need for QoS and SLA:**
- **➔ Self-Adapting Auto-Tuning systems**

Summary and Perspectives: On-going

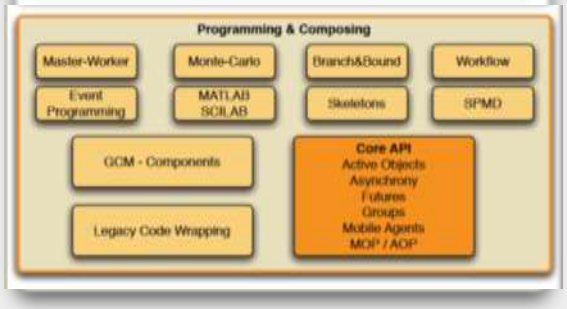


Conclusion: Currently Available

ProActive Parallel Suite

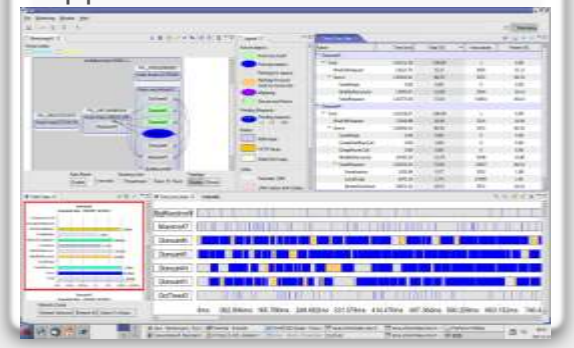
PROGRAMMING

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.

Further into the direction of:
Multi-Core + Distributed

Activeeon SCALE BEYOND LIMITS

ProActive Parallel Suite

OW2 Consortium

AGOS: Grid Architecture for SOA

Building a Platform for Agile SOA with Grid

- AGOS Solutions

In Open Source with Professional Support