Grand Large
INRIA

High Performance Computing on P2P Platforms: Recent Innovations

Franck Cappello
CNRS
Head Cluster et GRID group
INRIA Grand-Large
LRI, Université Paris sud.
fci@lri.fr
www.lri.fr/~fci

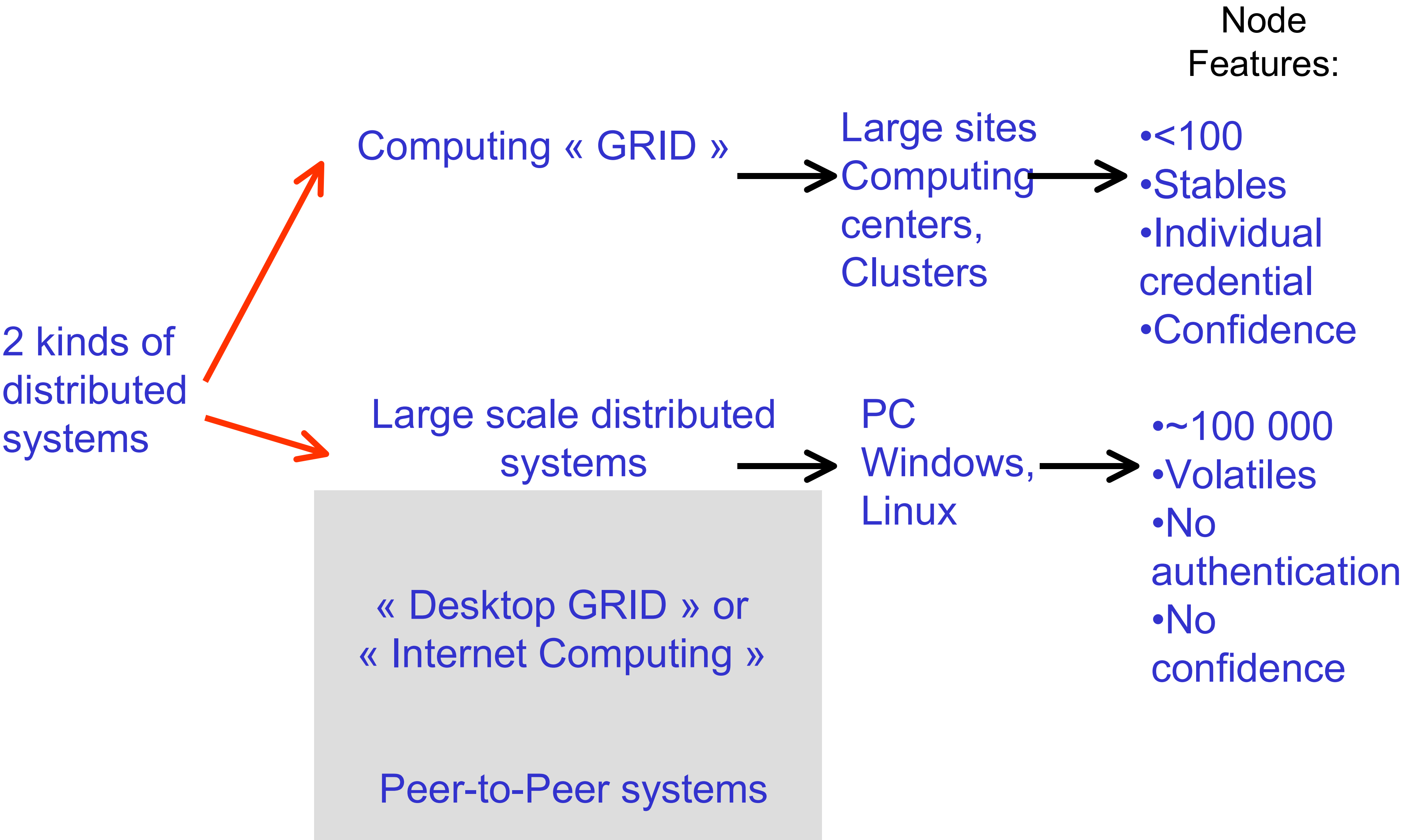
Action Concertée Incitative [ACI]
Globalisation des Ressources Informatiques
et des Données [GRID]

nference

Outline

- Introduction (GRID versus P2P)
- System issues in HPC P2P infrastructure
  - Internal of P2P systems for computing
  - Case Studies: XtremWeb / BOINC
- Programming HPC P2P infrastructures
  - RPC-V
  - MPICH-V (A message passing library For XtremWeb)
- Open issue: merging Grid & P2P
- Concluding remarks

Several types of GRID



Node Features:

2 kinds of distributed systems

Computing « GRID » → Large sites Computing centers, Clusters →
- <100
- Stables
- Individual credential
- Confidence

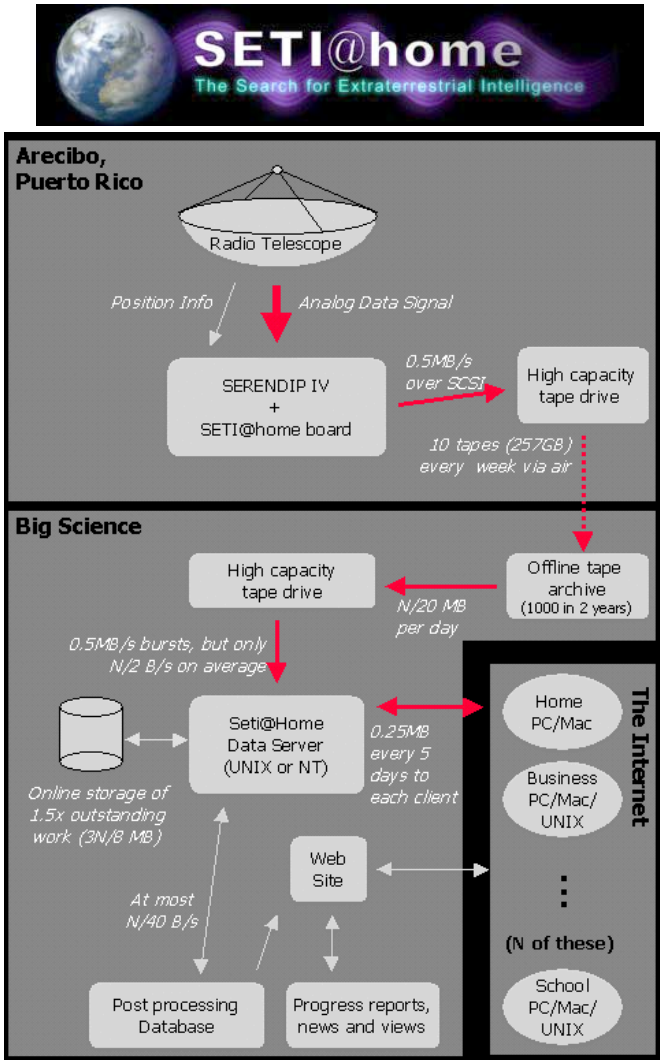
Large scale distributed systems → PC Windows, Linux →
- ~100 000
- Volatiles
- No authentication
- No confidence

« Desktop GRID » or « Internet Computing »

Peer-to-Peer systems

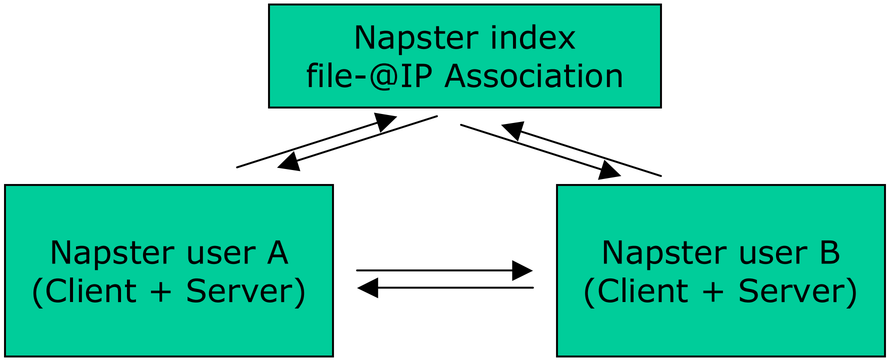
Large Scale Distributed Computing

- Principle
  - Millions of PCs
  - Cycle stealing
- Examples
  - SETI@HOME
    - Research for Extra Terrestrial I
    - 33.79 Teraflop/s (12.3 Teraflop/s for the ASCI White!)
  - DECRYPTHON
    - Protein Sequence comparison
  - RSA-155
    - Breaking encryption keys


SETI@home
The Search for Extraterrestrial Intelligence
Arecibo, Puerto Rico
Radio Telescope
Position Info
Analog Data Signal
SERENDIP IV + SETI@home board
0.5MB/s over SCSI
High capacity tape drive
10 tapes (257GB) every week via air
Big Science
Offline tape archive (1000 in 2 years)
N/20 MB per day
High capacity tape drive
0.5MB/s bursts, but only N/2 B/s on average
Seti@Home Data Server (UNIX or NT)
Online storage of 1.5x outstanding work (3N/8 MB)
0.25MB every 5 days to each client
Web Site
At most N/40 B/s
Post processing Database
Progress reports, news and views
The Internet
Home PC/Mac
Business PC/Mac/ UNIX
(N of these)
School PC/Mac/ UNIX

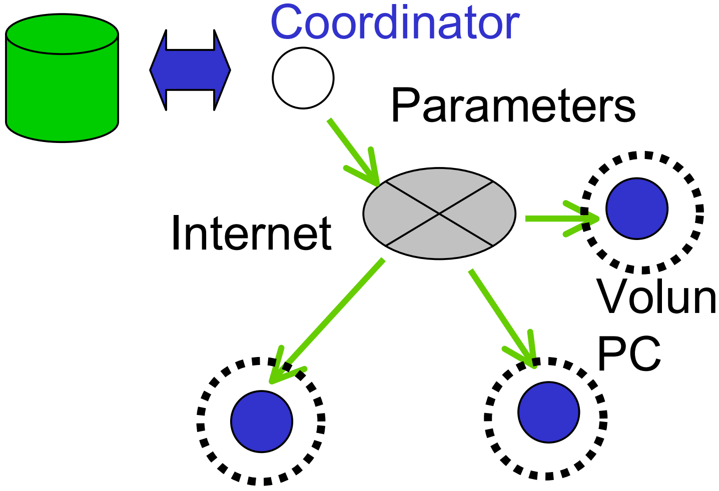
Large Scale P2P File Sharing

- Direct file transfer after index consultation
  - Client and Server issue direct connections
  - Consulting the index gives the client the @ of the server
- File storage
  - All servers store entire files
  - For fairness Client work as server too.
- Data sharing
  - Non mutable Data
  - Several copies no consistency check
- Interest of the approach
  - Proven to scale up to million users
  - Resilience of file access
- Drawback of the approach
  - Centralized index
  - Privacy violated



Distributed Computing

A central coordinator schedules tasks on volunteer computers,
Master worker paradigm,
Cycle stealing



Client application
Params. /results.

Coordinator

Parameters

Internet

Volunteer PC

Volunteer PC

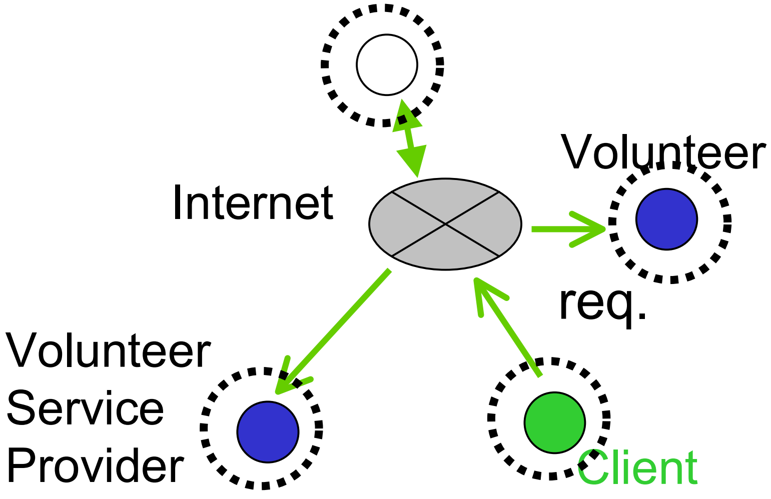
Volunteer PC
Downloads and executes the application

- Dedicated Applications
  - SETI@Home, distributed.net,
  - Décrypthon (France)
- Production applications
  - Folding@home, Genome@home,
  - Xpulsar@home,Folderol,
  - Exodus, Peer review,
- Research Platforms
  - Javelin, Bayanihan, JET,
  - Charlotte (based on Java),
- Commercial Platforms
  - Entropia, Parabon,
  - United Devices, Platform (AC)

Peer to Peer systems (P2P)

All system resources
-may play the roles of client
and server,
-may communicate directly
Distributed and self-organizing
infrastructure

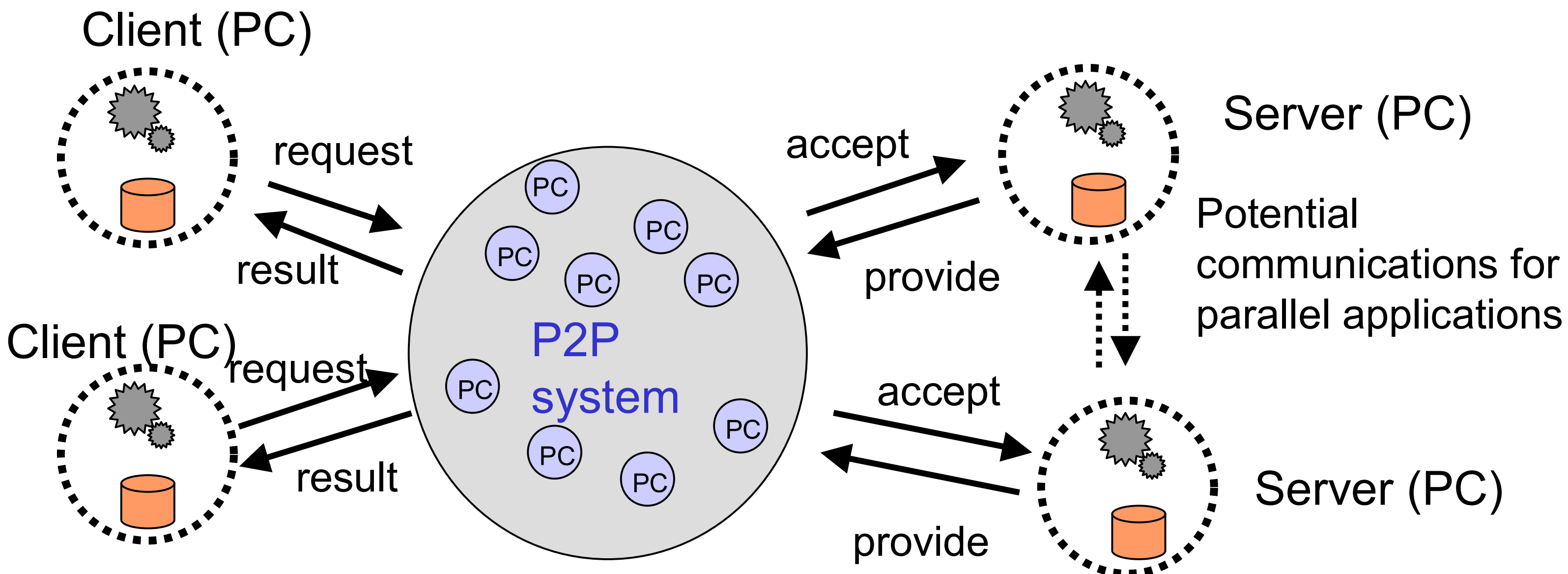
Volunteer PC
participating to the resource
discovery/coordination



- User Applications
  - Instant Messaging
  - Managing and Sharing Information
  - Collaboration
  - Distributed storage
- Middleware
  - Napster, Gnutella, Freenet,
  - KaZaA, Music-city,
  - Jabber, Groove,
- Research Projects
  - Globe (Tann.), Cx (Javalin), Farsite,
  - OceanStore (USA),
  - Pastry, Tapestry/Plaxton, CAN, Chord,
- Other projects
  - Cosm, Wos, peer2peer.org,
  - JXTA (sun), PtPTL (intel),

Merging Internet & P2P Systems: P2P Distributed Computing

Allows any node to play different roles (client, server, system infrastructure)



Potential communications for parallel applications

Request may be related to Computations or data

Accept concerns computation or data

A very simple problem statement but leading to a lot of research issues:

scheduling, security, message passing, data storage

Large Scale enlarges the problematic: volatility, confidence, etc.

"Three Obstacles to Making P2P Distributed Computing Routine"

1) New approaches to problem solving
   - Data Grids, distributed computing, peer-to-peer, collaboration grids, …
2) Structuring and writing programs
   - Abstractions, tools
3) Enabling resource sharing across distinct institutions
   - Resource discovery, access, rese authentication, authorization, policy, communication, fault detection and notification; …

Programming Problem

Systems Problem

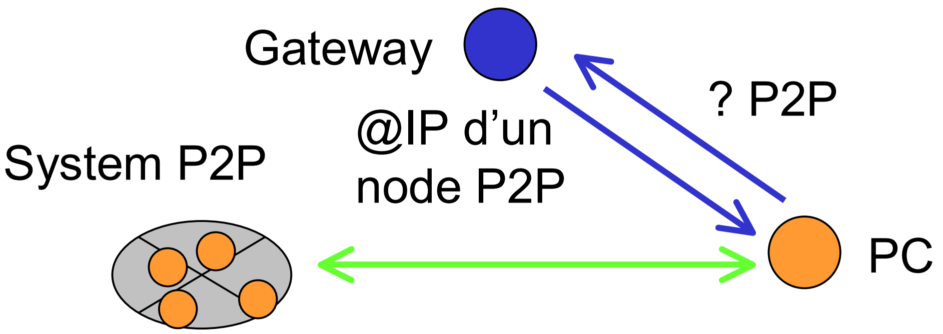
Credit: Ian Foster

Outline

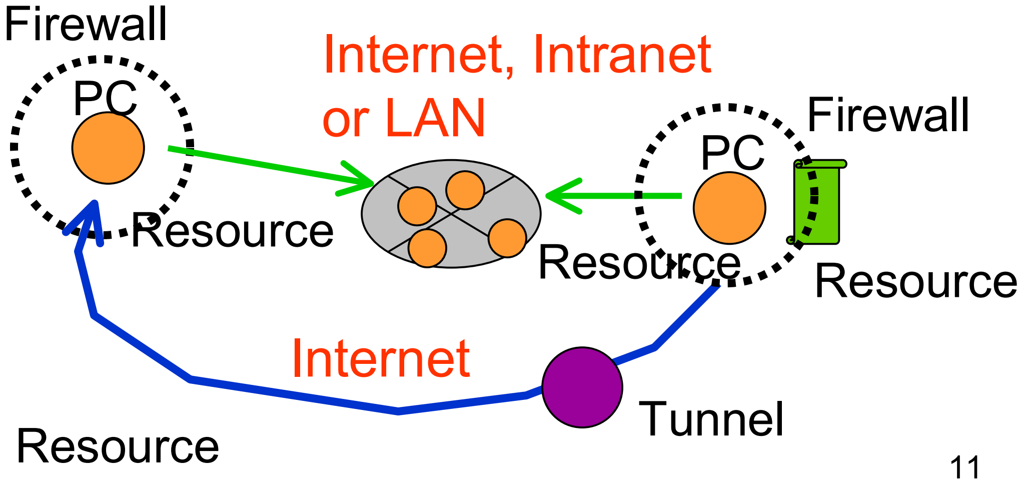
- Introduction (large scale distributed systems)
- System issues in HPC P2P infrastructure
  - Internal of P2P systems for computing
  - Case Studies: XtremWeb / BOINC
- Programming HPC P2P infrastructures
  - RPC-V
  - MPICH-V (A message passing library For XtremWeb)
- Open issue: merging Grid & P2P
- Concluding remarks

Basic components of P2P systems

1) Gateway (@IP, Web pages, etc.)
-Give the @ of other nodes
-Choose a community,
-Contact a community manager

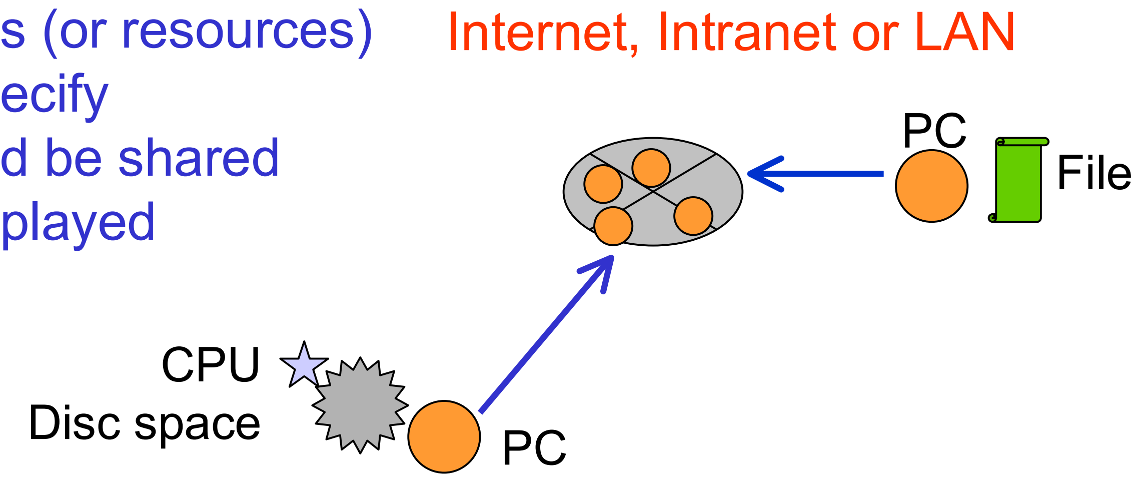


2) Connection/Transport protocol for requests, results and control
-Bypass firewalls,
-Build a virtual address space (naming the participants: NAT)
(Tunnel, push-pull protocols)



Basic components of P2P systems

3) Publishing services (or resources)
Allows the user to specify
-what resources could be shared
-what roles could be played
-what protocol to use
(WSDL, etc.)



Internet, Intranet or LAN

PC

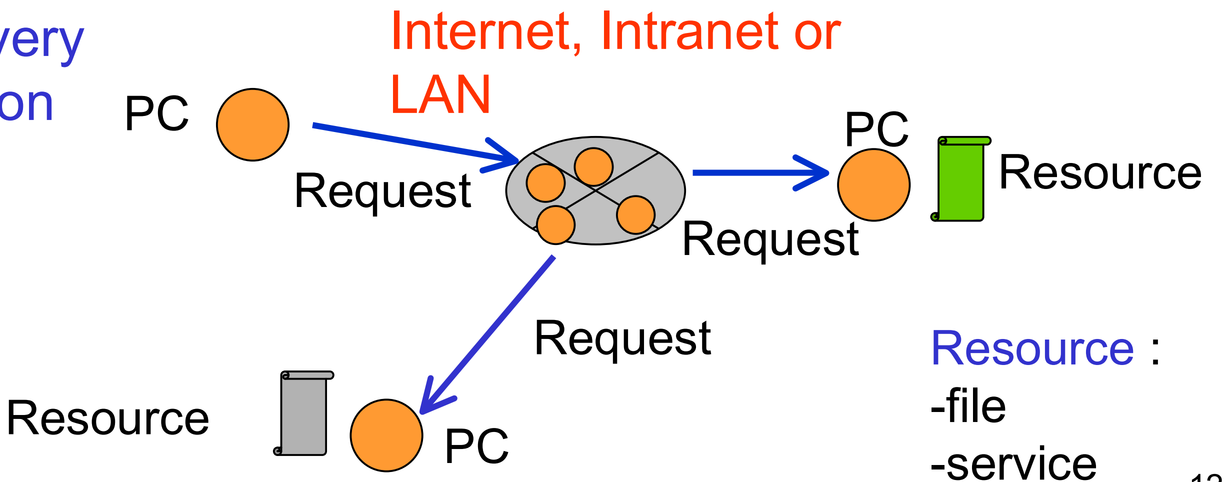
File

CPU

Disc space

PC

4) Resource discovery
(establish connection
between client and
service providers)
(Centralized directory,
hierarchical
directory, flooding,
search in topology)



Internet, Intranet or LAN

PC

Request

PC

Resource

Request

Request

Resource

PC

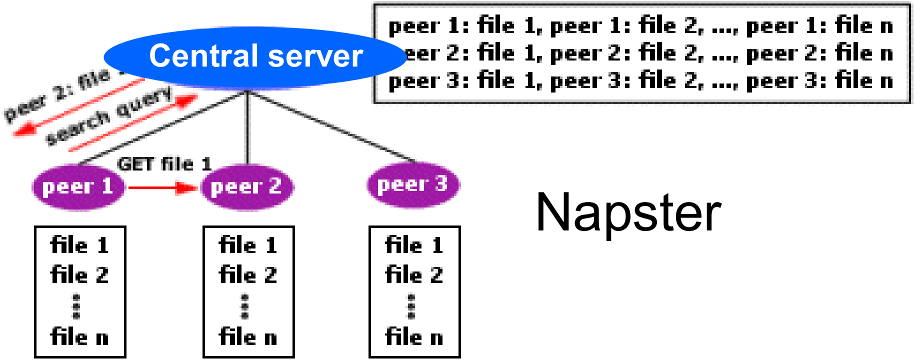
Resource :
-file
-service

Grand Large

Resource Discovery in P2P Systems

1st Generation:

Central index

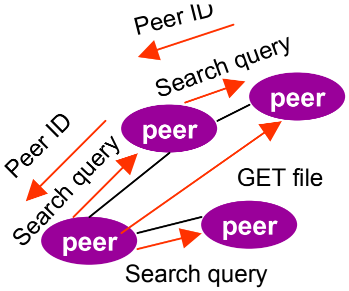


peer 1: file 1, peer 1: file 2, ..., peer 1: file n
peer 2: file 1, peer 2: file 2, ..., peer 2: file n
peer 3: file 1, peer 3: file 2, ..., peer 3: file n

Napster

2nd Generation:

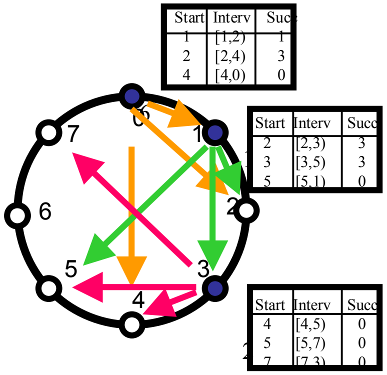
No central server:
Flooding



Gnutella,

3rd Generation:

Distributed Hash Table
(self organizing overlay
network: topology, routing)



Start	Interv	Suc
1	[1,2)	1
2	[2,4)	3
4	[4,0)	0

Start	Interv	Succ
2	[2,3)	3
3	[3,5)	3
5	[5,1)	0

Start	Interv	Succ
4	[4,5)	0
5	[5,7)	0
7	[7,3)	0

CAN, Chord,
Pastry, etc.

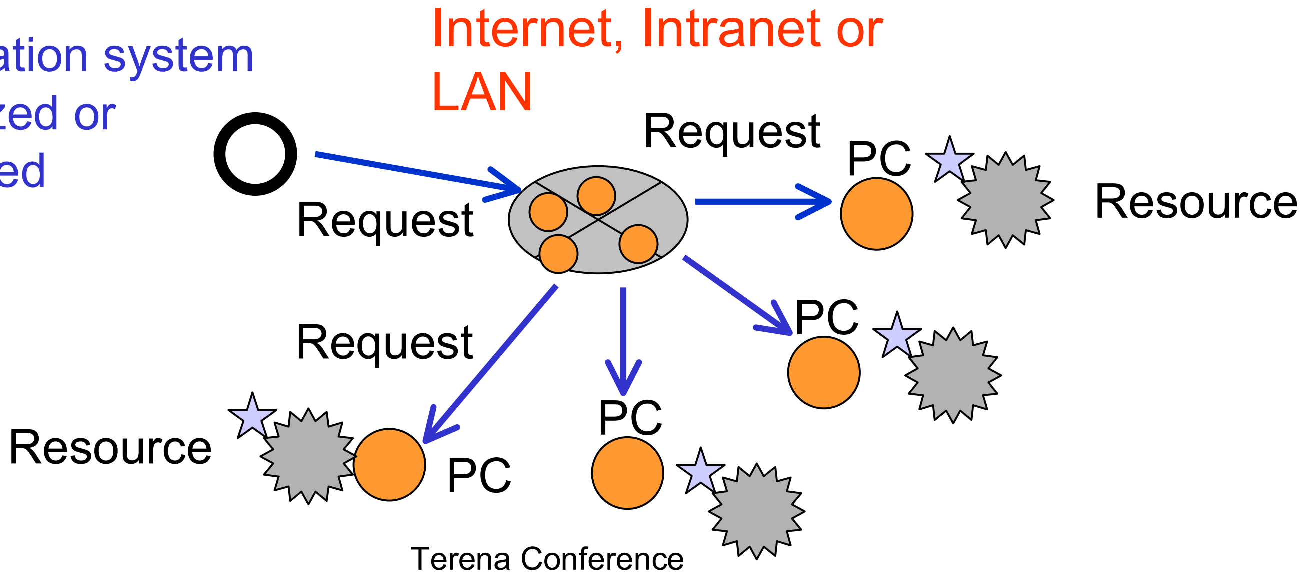
Additional component of P2P systems for Computing

The role of the 4 previous components was A) to setup the system and B) to discover a set of resources for a client

5) Coordination sys.: (virtual cluster manager)

- Receives Client computing request
- Configures/Manages a platform (collect service proposals and attribute roles)
- Schedules tasks / data distribution-transfers
- Detects/recovers Faults

Coordination system Centralized or Distributed

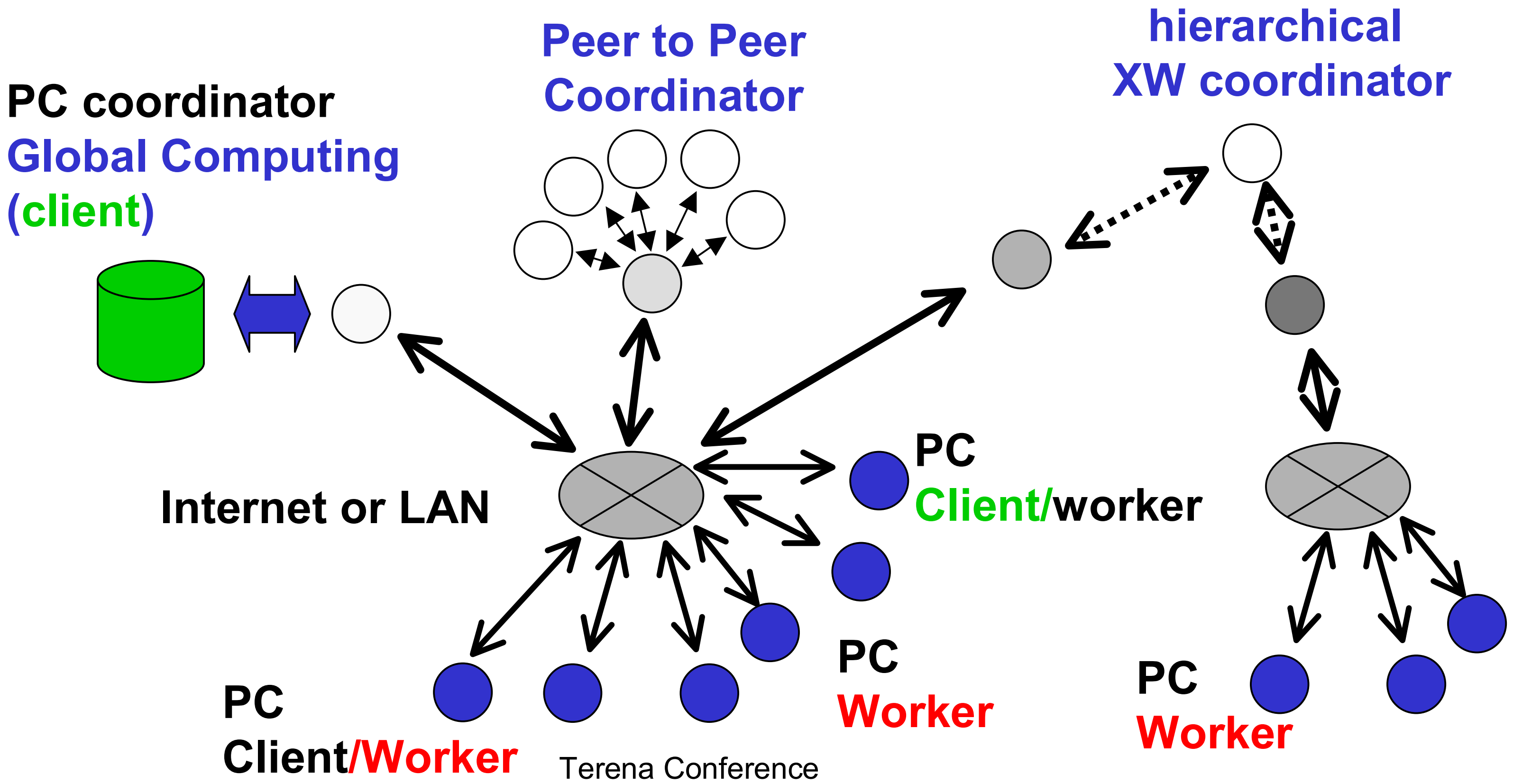


Outline

- Introduction (large scale distributed systems)
- System issues in HPC P2P infrastructure
  - Internal of P2P systems for computing
  - Case Studies: XtremWeb / BOINC
- Programming HPC P2P infrastructures
  - RPC-V
  - MPICH-V (A message passing library For XtremWeb)
- Open issue: merging Grid & P2P
- Concluding remarks

XtremWeb: General Architecture

- XtremWeb 1 implements a subset of the 5 P2P components
- 3 entities : client/coordinator/worker (diff protect. domains)
- Current implementation: centralized coordinator



PC coordinator
Global Computing
(client)

Peer to Peer
Coordinator

hierarchical
XW coordinator

Internet or LAN

PC
Client/worker

PC
Client/Worker

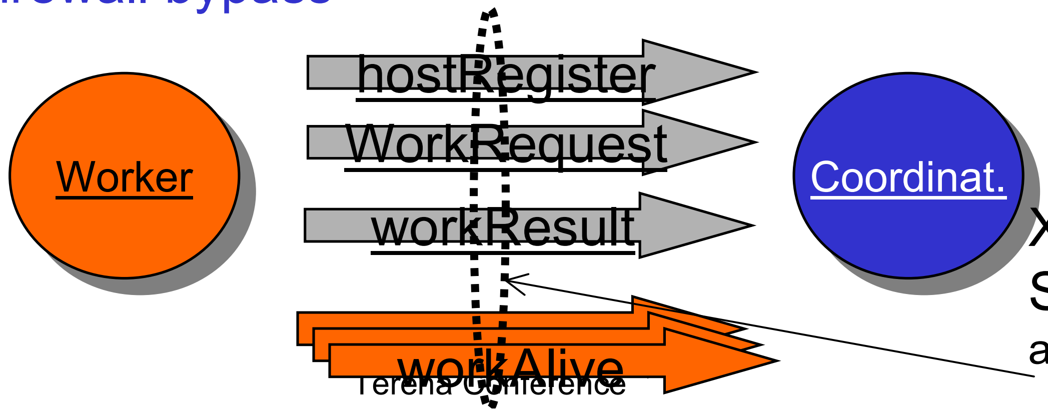
PC
Worker

PC
Worker

XW: Worker Architecture

Applications	→ Binary (legacy codes CHP en Fortran ou C) → Java (recent codes, object codes)
OS	→ Linux, SunOS, Mac OSX, → Windows
Auto-monitoring	→ Trace collection

Protocol : firewall bypass



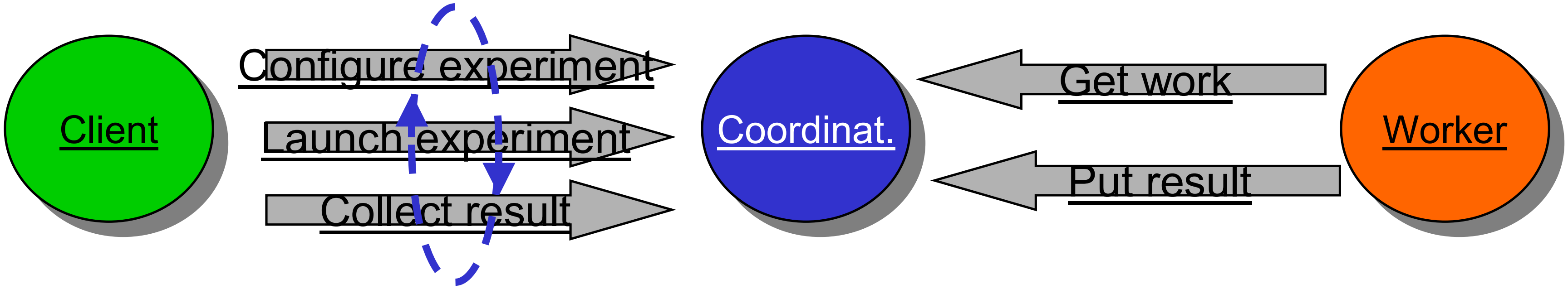
XML RPC et
SSL authentication
and encryption

XW: Client architecture

A API Java → XWRPC
→ task submission
→ result collection
→ Monitoring/control

Bindings →OmniRPC, GridRPC

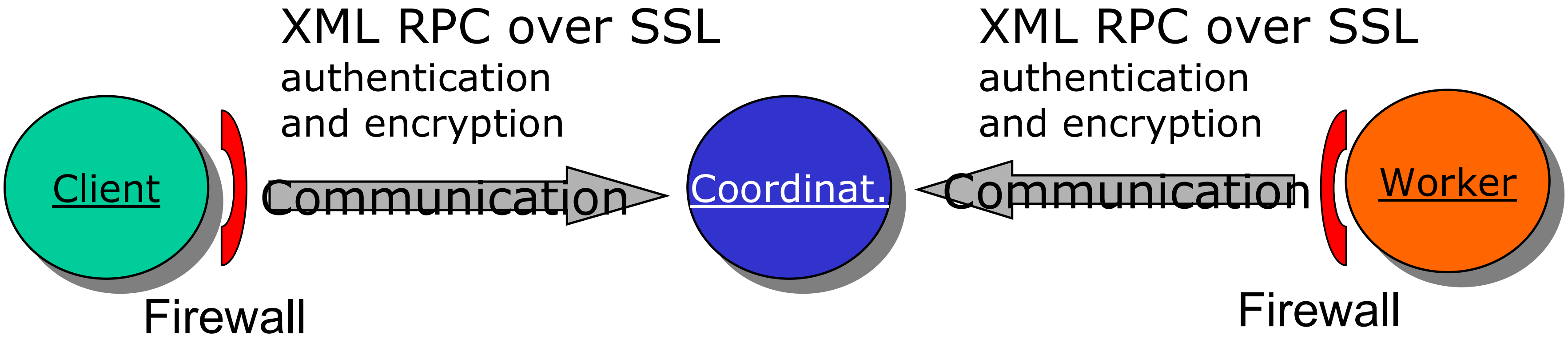
Applications →Multi-parameter, bag of tasks
→Master-Wroker (iterative), EP



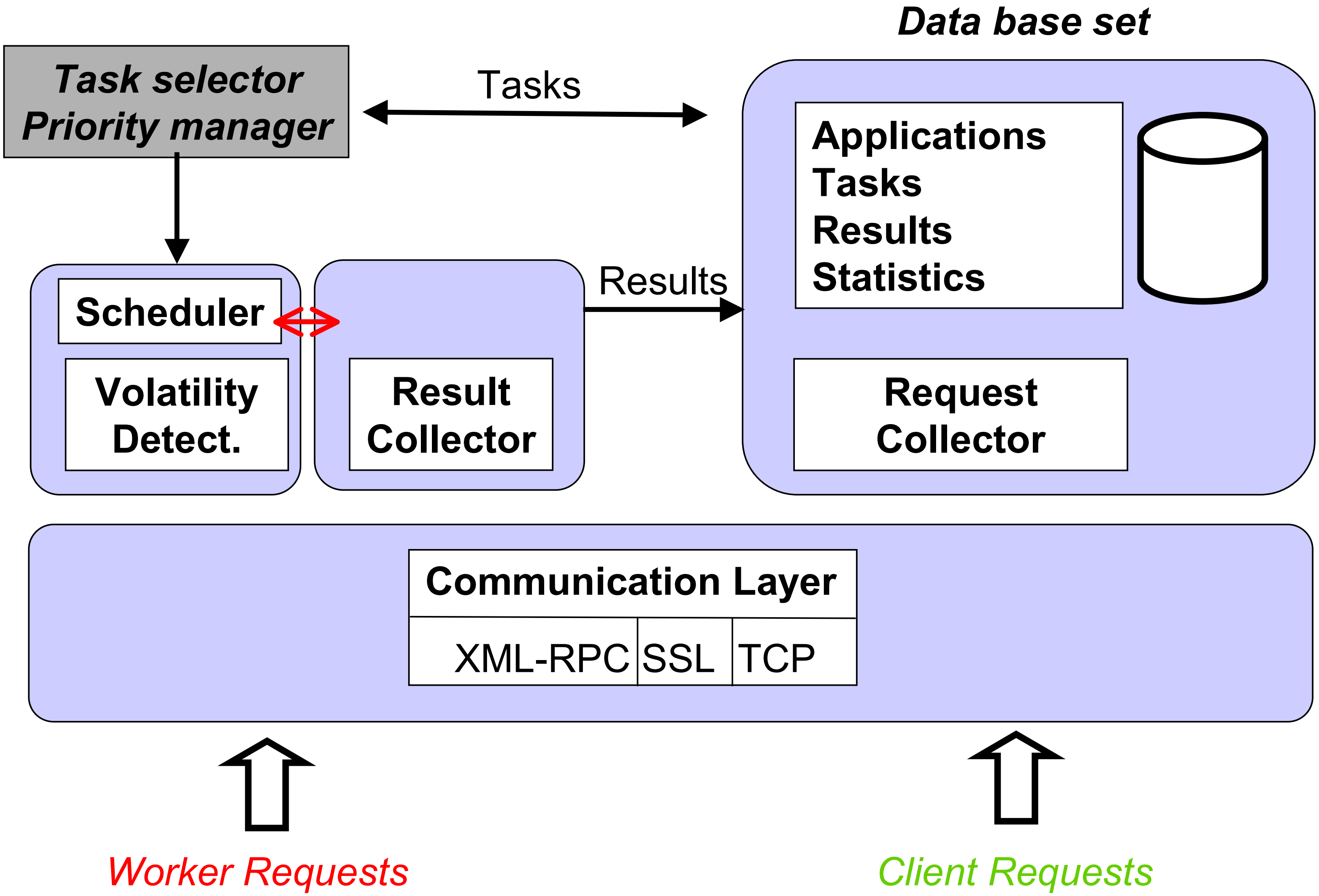
XW: Security model

Firewall bypass:

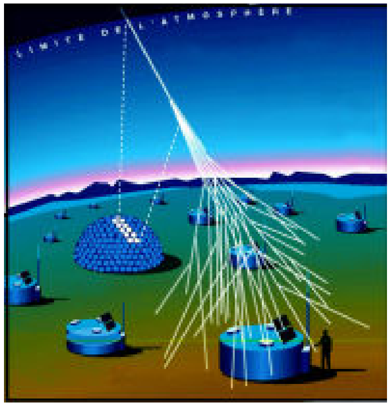
- → Sandboxing (SBLSM) + action logging on worker and coordinator
- → Client Authentication for Coordinator access (Public/Private key)
- → Communication encryption between all entities
- → Coordinator Authentication for Worker access *(Public/Private Key)*
- → *Certificate + certificate authority*



XW: coordinator architecture

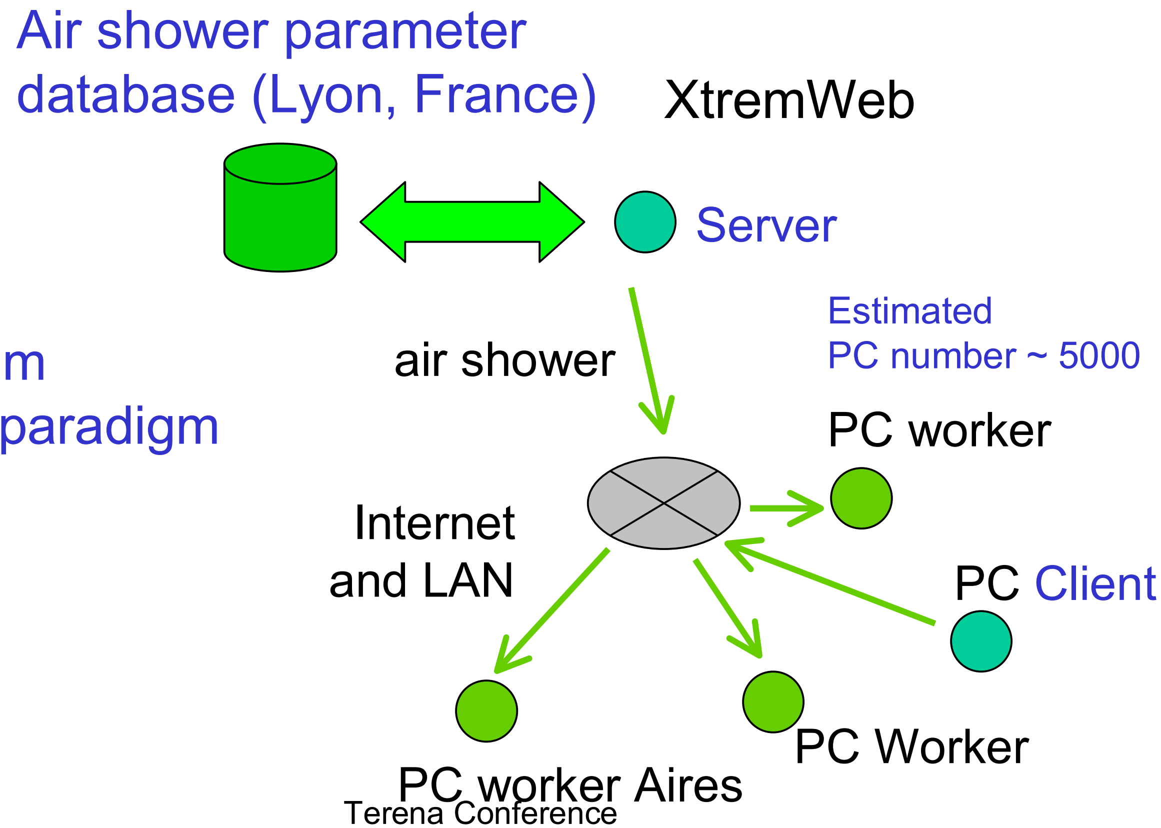


XtremWeb Application: Pierre Auger Observatory



Understanding the origin of very high cosmic rays:

- Aires: Air Showers Extended Simulation
  - Sequential, Monte Carlo. Time for a run: 5 to 10 hours (500MhzPC)



Air shower parameter database (Lyon, France)

XtremWeb

Server

air shower

Estimated PC number ~ 5000

PC worker

Internet and LAN

PC Client

PC Worker

PC worker Aires

- Trivial parallelism
- Master Worker paradigm

Deployment example

Application :

AIRES (Auger)

Deployment:

- Coordinator at LRI
- Madison: 700 workers

  Pentium III, Linux

  (500 MHz+933 MHz)

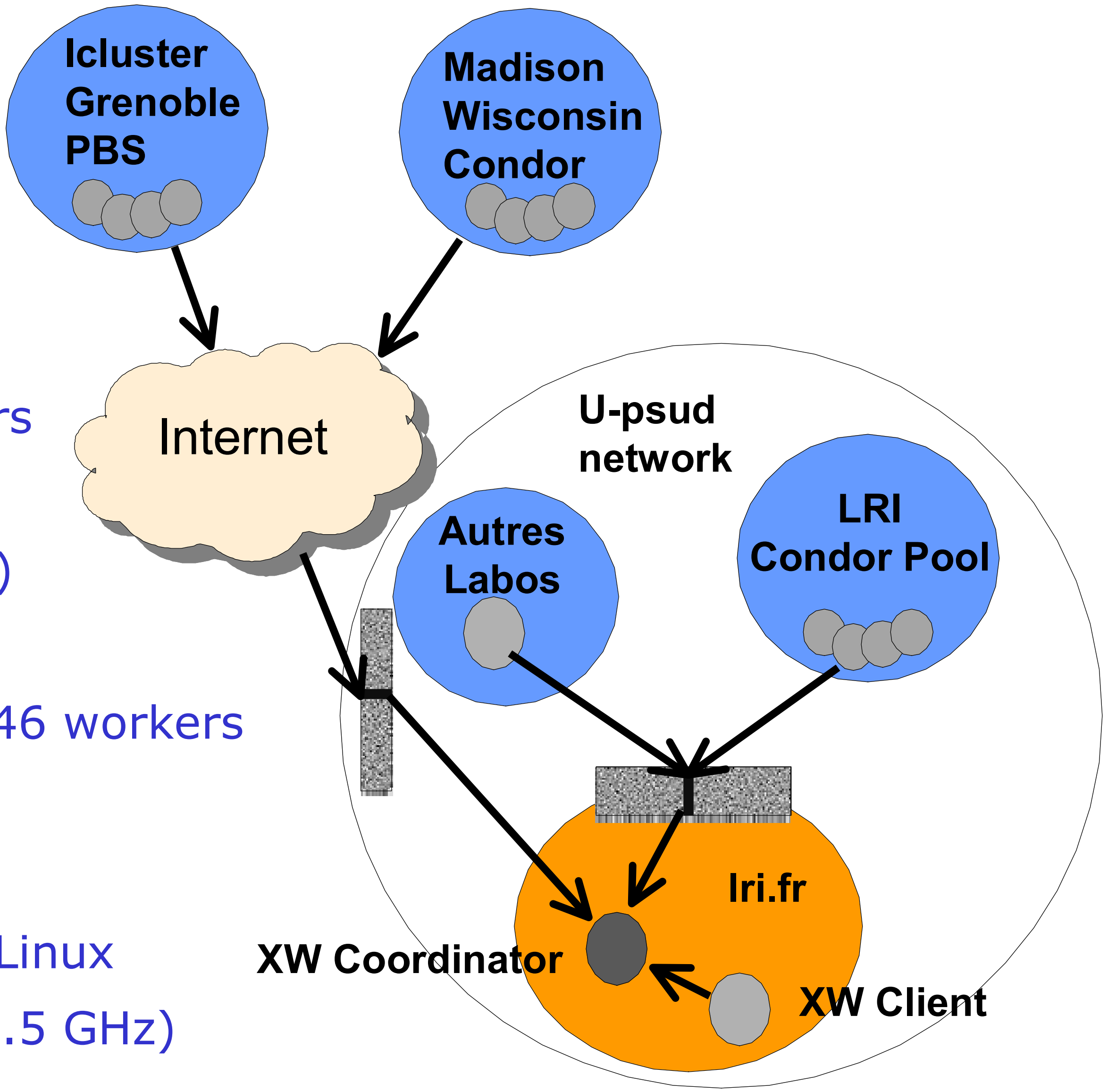
  (Condor pool)
- Grenoble Icluster: 146 workers

  (733 Mhz), PBS
- LRI: 100 workers

  Pentium III, Athlon, Linux

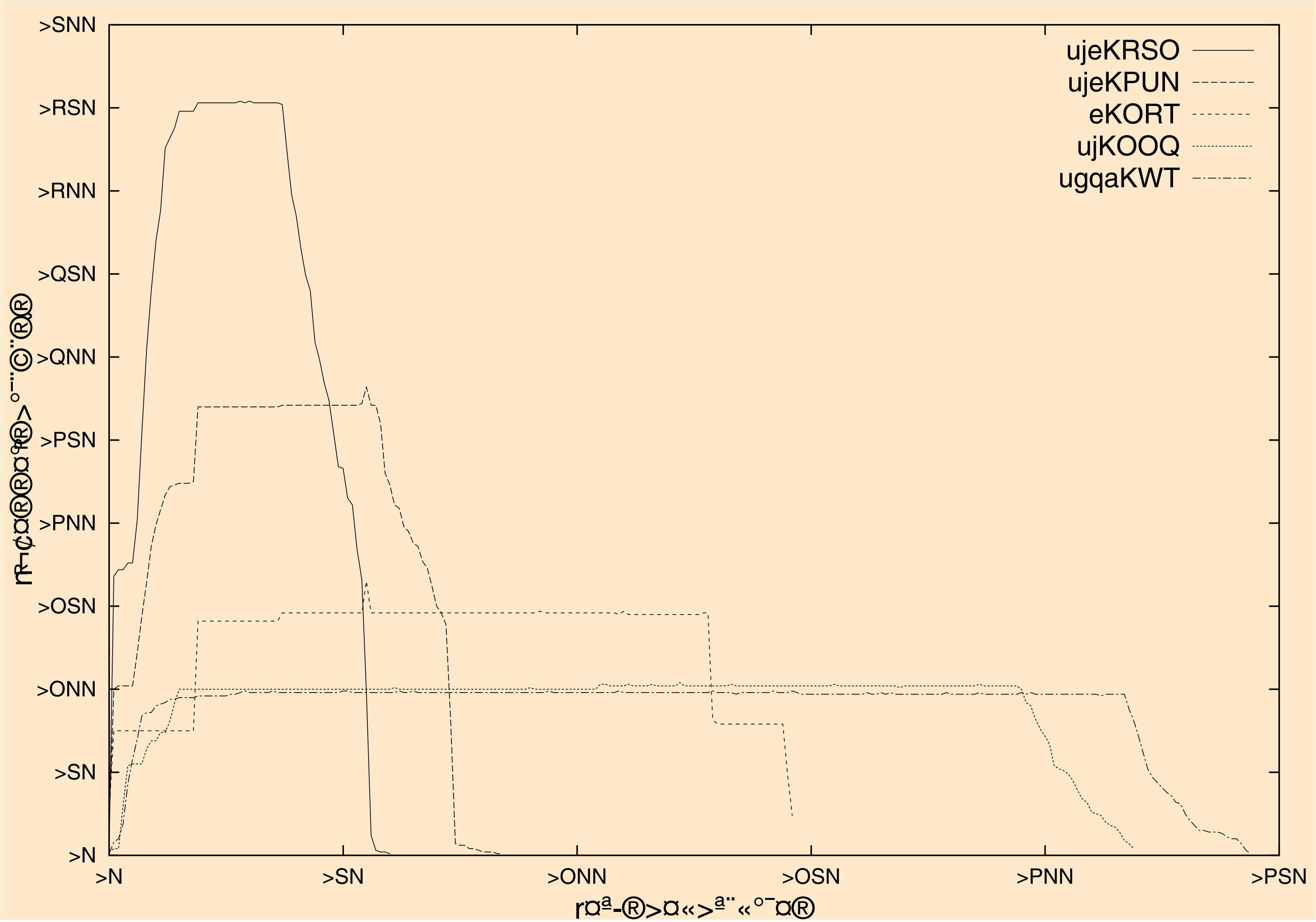
  (500MHz, 733MHz, 1.5 GHz)

  (Condor pool)

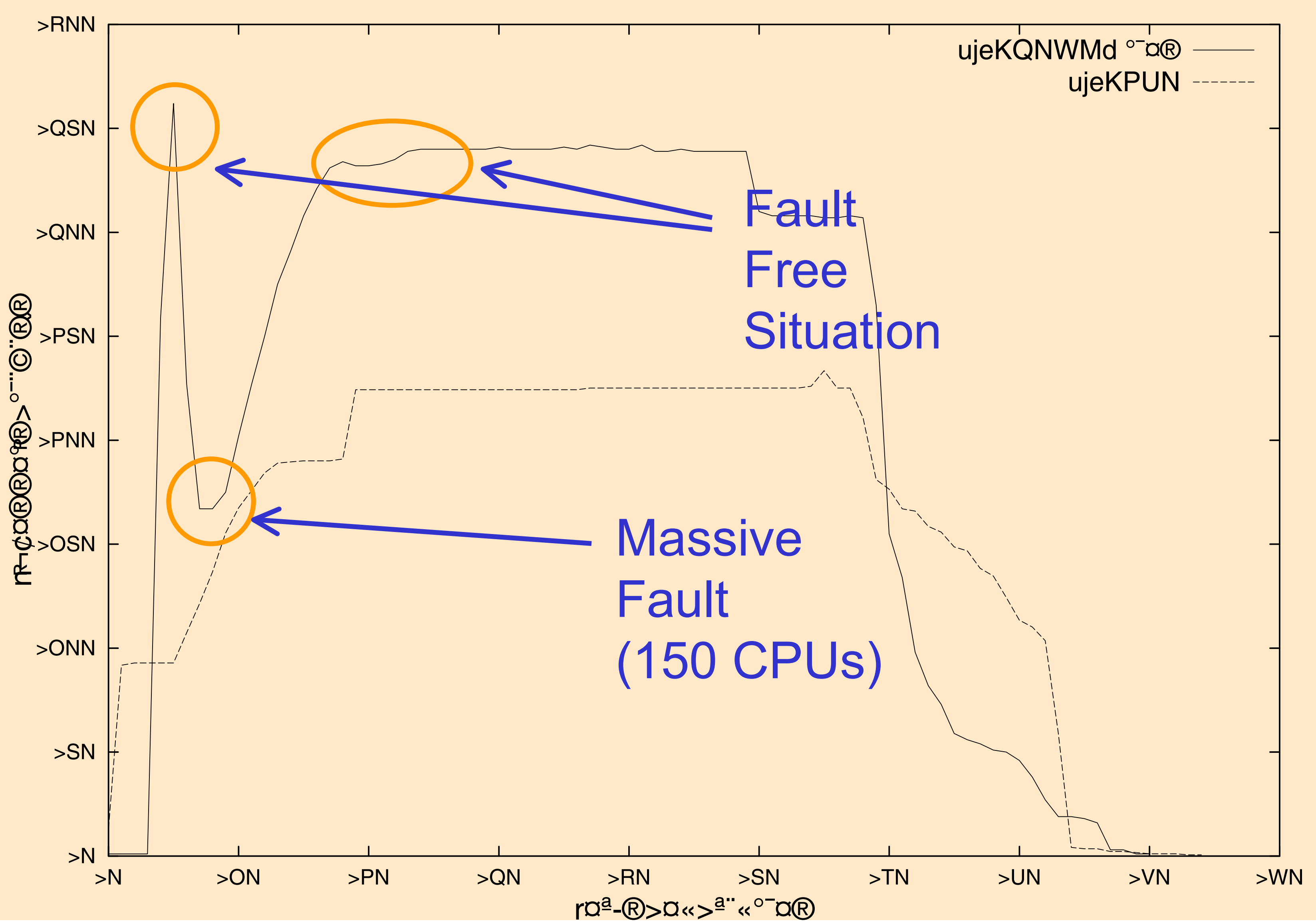


Grand Large

XtremWeb for AIRES



Auger-XW (AIRES): High Energy Physics



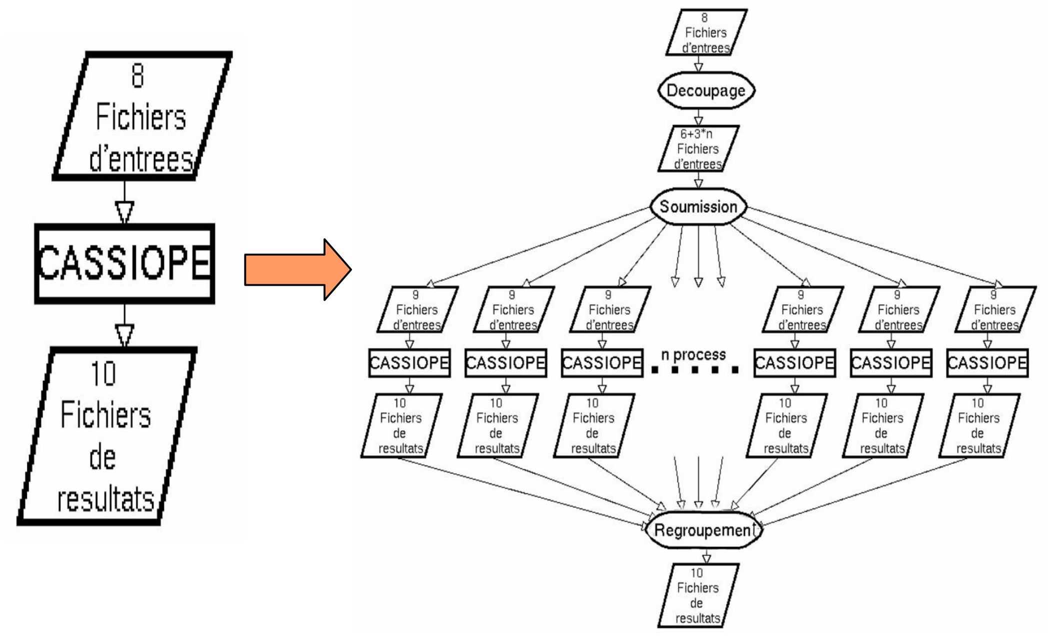
Fault
Free
Situation

Massive
Fault
(150 CPUs)

Grand Large

EADS-CCR (Airbus, Ariane)

Cassiope application: Ray-tracing


8 Fichiers d'entrees

CASSIOPE

10 Fichiers de resultats

8 Fichiers d'entrees

Decoupage

6+3*n Fichiers d'entrees

Soumission

9 Fichiers d'entrees

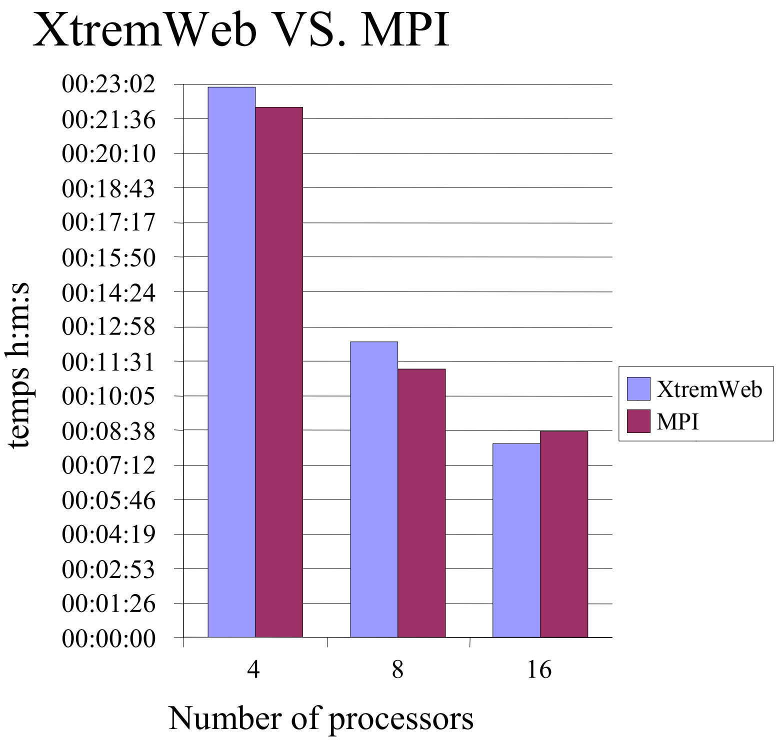
CASSIOPE

n process

10 Fichiers de resultats

Regroupement

10 Fichiers de resultats



XtremWeb: User projects

1 **CGP2P ACI GRID** (academic research on Desktop Grid systems), France

2 C@sper **Industry research project (Airbus + Alcatel Space),** France

3 **Augernome XtremWeb** (Campus wide Desktop Grid), France

4 **EADS (Airplane + Ariane rocket manufacturer),** France

5 **IFP (French Petroleum Institute),** France

6 **University of Geneva**, (research on Desktop Grid systems), Switzerland

7 **University of Winsconsin Madisson,** Condor+XW, USA

8 **University of Gouadeloupe + Paster Institute:** Tuberculoses, France

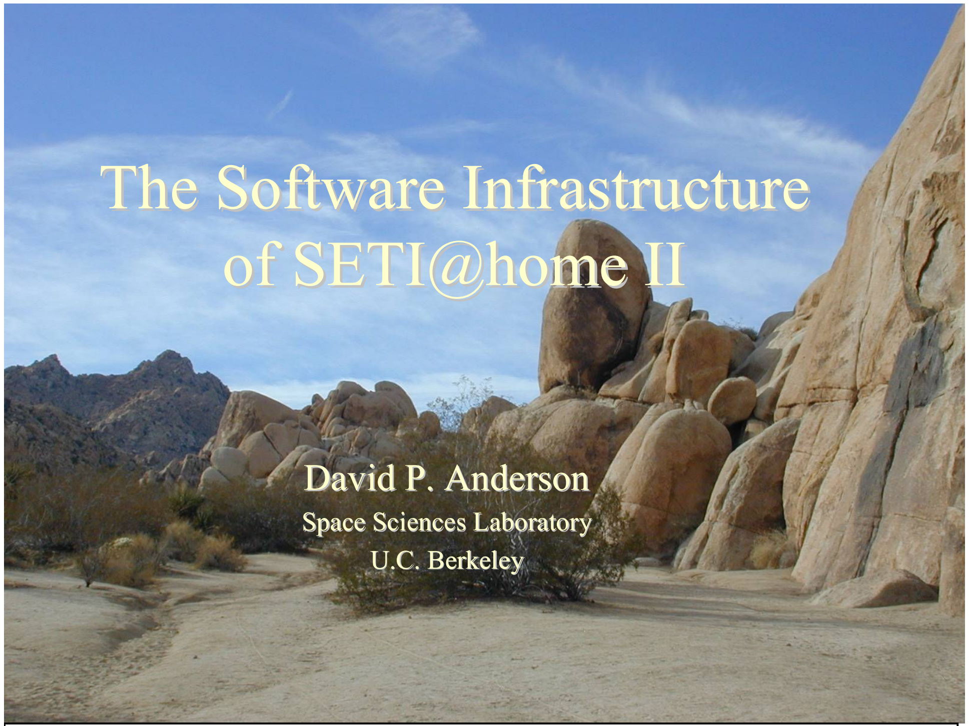
9 **Mathematics lab** University of Paris South (PDE solver research) , France

10 **University of Lille** (control language for Desktop Grid systems), France

11 **ENS Lyon**: research on large scale storage, France

12 **IRISA (INRIA Rennes),**

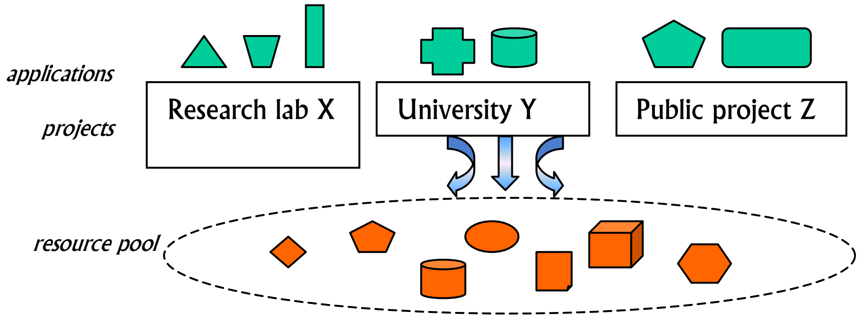
13 CEA Saclay



The Software Infrastructure of SETI@home II

David P. Anderson
Space Sciences Laboratory
U.C. Berkeley

Goals of a PRC platform



- Participants install one program, select projects, specify constraints; all else is automatic
- Projects are autonomous
- Advantages of a shared platform:
  - Better instantaneous resource utilization
  - Better resource utilization over time
  - Faster/cheaper for projects, software is better
  - Easier for projects to get participants
  - Participants learn more

Distributed computing platforms

- Academic and open-source
  - Globus
  - Cosm
  - XtremWeb
  - Jxta
- Commercial
  - Entropia
  - United Devices
  - Parabon

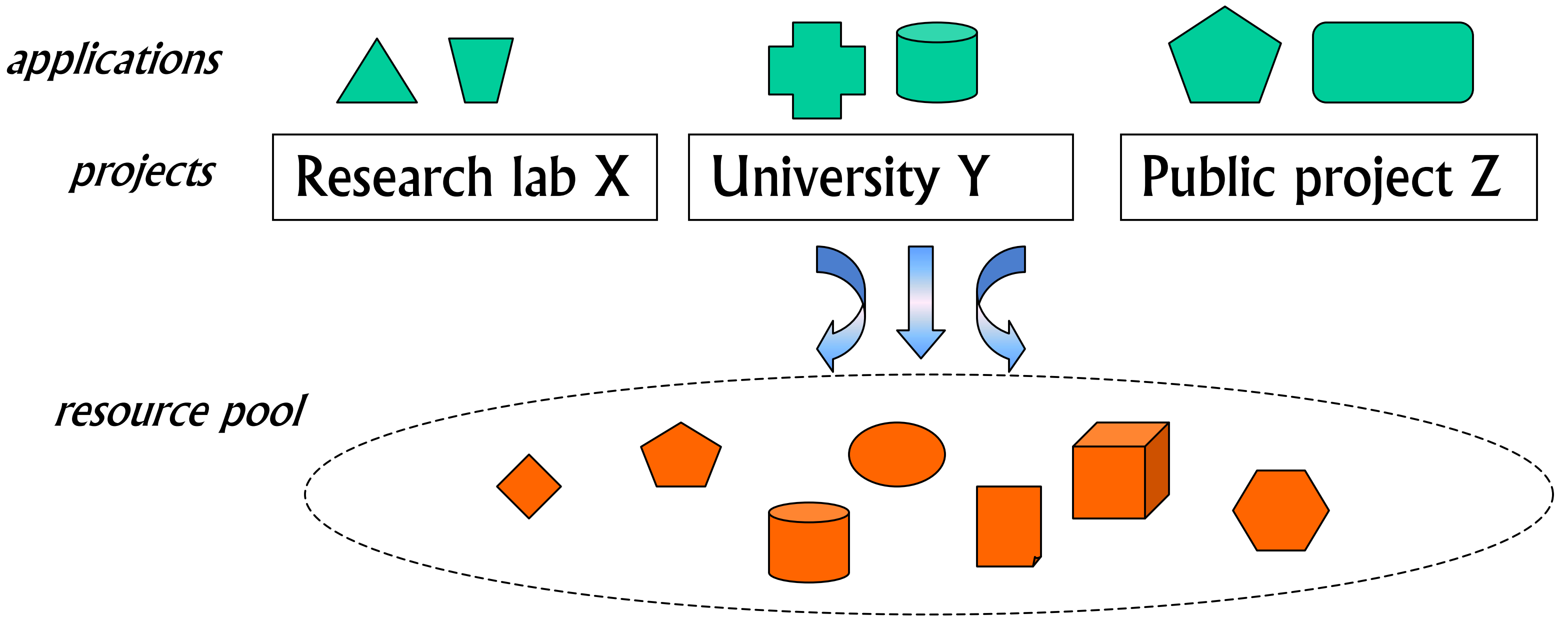
Goals of BOINC

(Berkeley Open Infrastructure for Network Computing)

- Public-resource computing/storage
- Multi-project, multi-application
  - Participants can apportion resources
- Handle fairly diverse applications
- Work with legacy apps
- Support many participant platforms
- Small, simple

Credit: David Anderson

BOINC: Berkeley Open Infrastructure for Network Computing

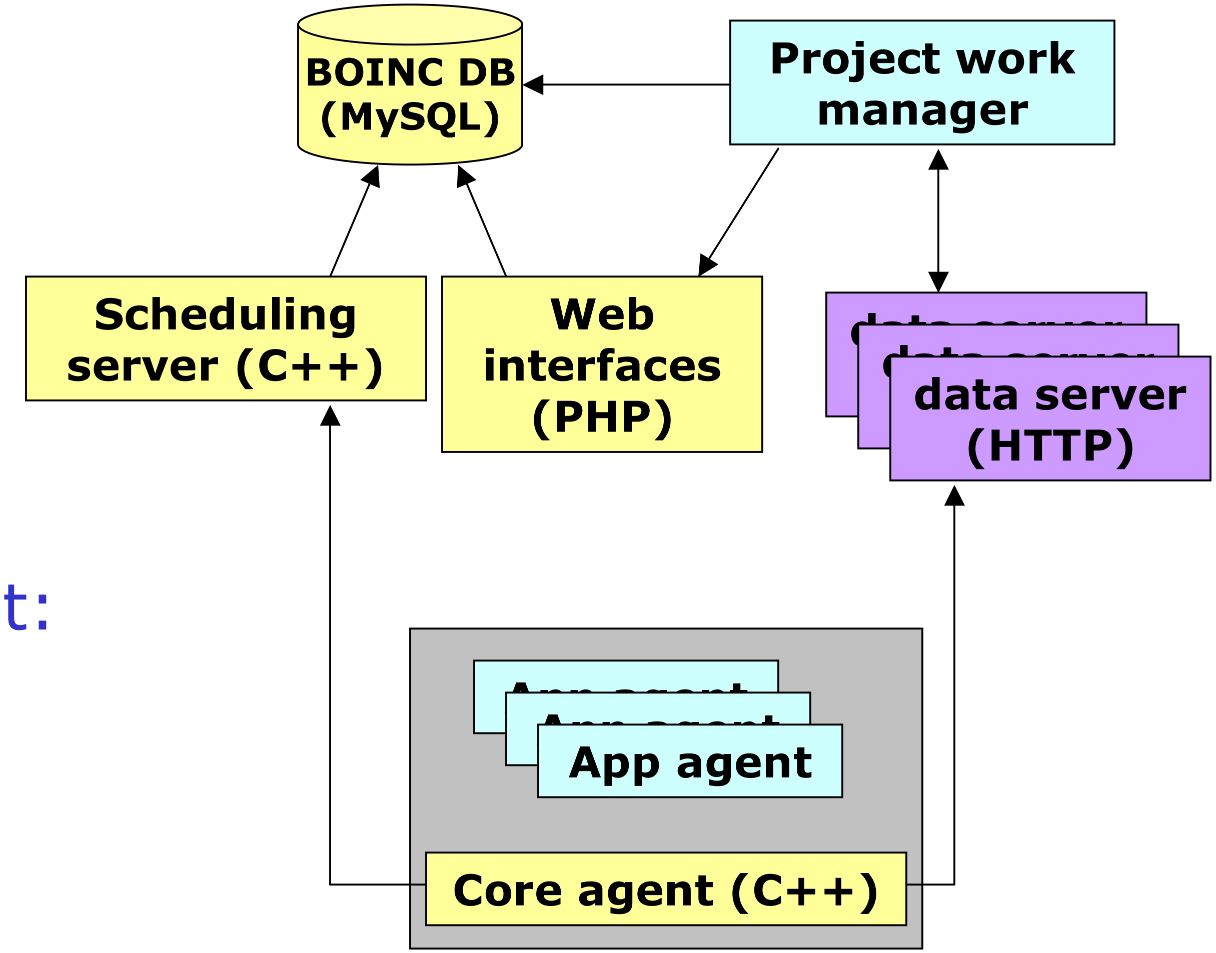


- Multiple autonomous projects
- Participants select projects, allocate resources
- Support for data-intensive applications
- Redundant computing, credit system

Credit: David Anderson

Anatomy of a BOINC project

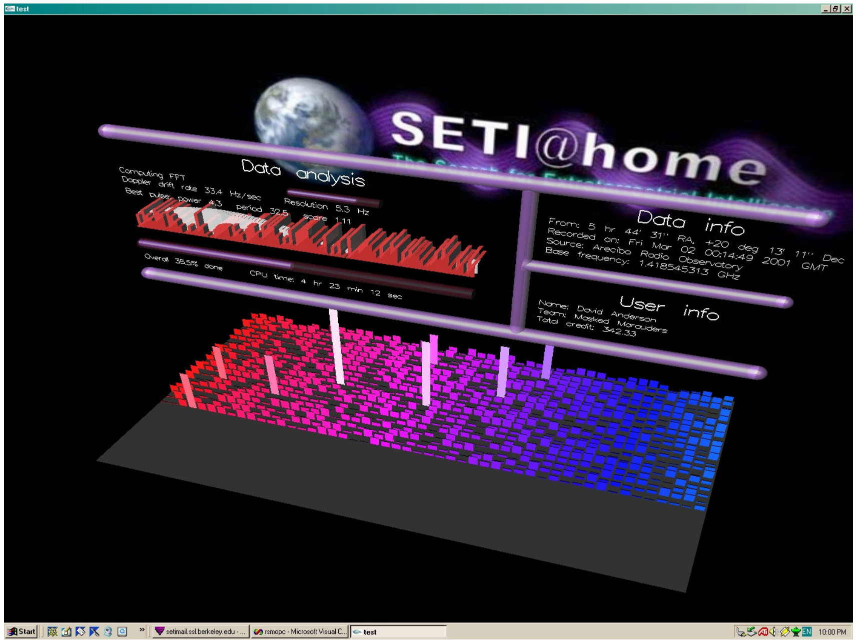
- Project:
- Participant:



Credit: David Anderson

BOINC Applications

- Applications (EP):
  - SETI@home I and II
  - Astropulse
  - Folding@home?
  - Climateprediction.net?
- Status:
  - NSF funded
  - In beta test
  - See http://boinc.berkeley.edu



Data analysis
Computing FFT
Doppler drift rate 33.4 Hz/sec
Resolution 5.3 Hz
Best pulse: power 4.3 period 32.5 score 1.11
Overall 35.5% done
CPU time: 4 hr 23 min 12 sec

Data info
From: 5 hr 44' 31'' RA, +20 deg 13' 11'' Dec
Recorded on: Fri Mar 02 00:14:49 2001 GMT
Source: Arecibo Radio Observatory
Base frequency: 1.418545313 GHz

User info
Name: David Anderson
Team: Masked Marauders
Total credit: 342.33

Credit: David Anderson

User Feedback

Deployment is a complex issue:

- → Human factor (system administrator, PC owner)
- → Installation on a case to case basis
- → Use of network resources (backup during the night)
- → Dispatcher scalability (hierarchical, distributed?)
- → Complex topology (NAT, firewall, Proxy).

Computational resource capacities limit the application range:

- → Limited memory (128 MB, 256 MB),
- → Limited network performance (100baseT),

Lack of programming models limit the application port:

- → Need for RPC
- → Need for MPI

Users don’t understand immediately the available computational power

- → When they understand, they propose new utilization of their applications (similar to the transition from sequential to parallel) They also rapidly ask for more resources!!!

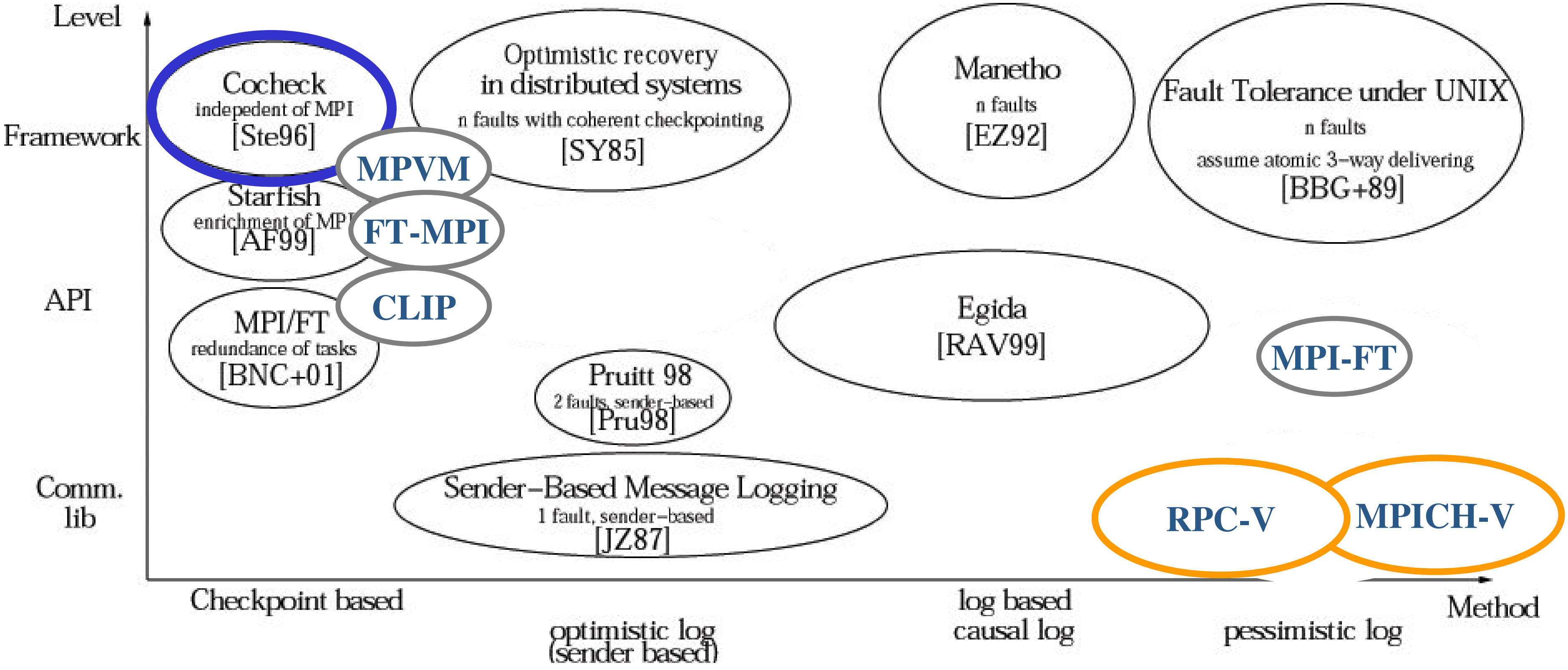
Strong need for tools helping users browsing the massive amount of results

Outline

- Introduction (large scale distributed systems)
- System issues in HPC P2P infrastructure
  - Internal of P2P systems for computing
  - Case Studies: XtremWeb / BOINC
- Programming HPC P2P infrastructures
  - RPC-V
  - MPICH-V (A message passing library For XtremWeb)
- Open issue: merging Grid & P2P
- Concluding remarks

Fault tolerant explicit message passing: Related work

A classification of fault tolerant message passing libraries
considering A) level in the software stack where fault tolerance is managed and
B) fault tolerance techniques.



RPC-V (Volatile)

Goal: execute RPC like applications on volatile nodes

Programmer’s view unchanged:

PC client RPC(Foo, params.)

PC Server Foo(params.)

Problems: 1) volatile nodes (any number at any time)
2) firewalls (PC Grids)
3) Recursion (recursive RPC calls)

Objective summary:

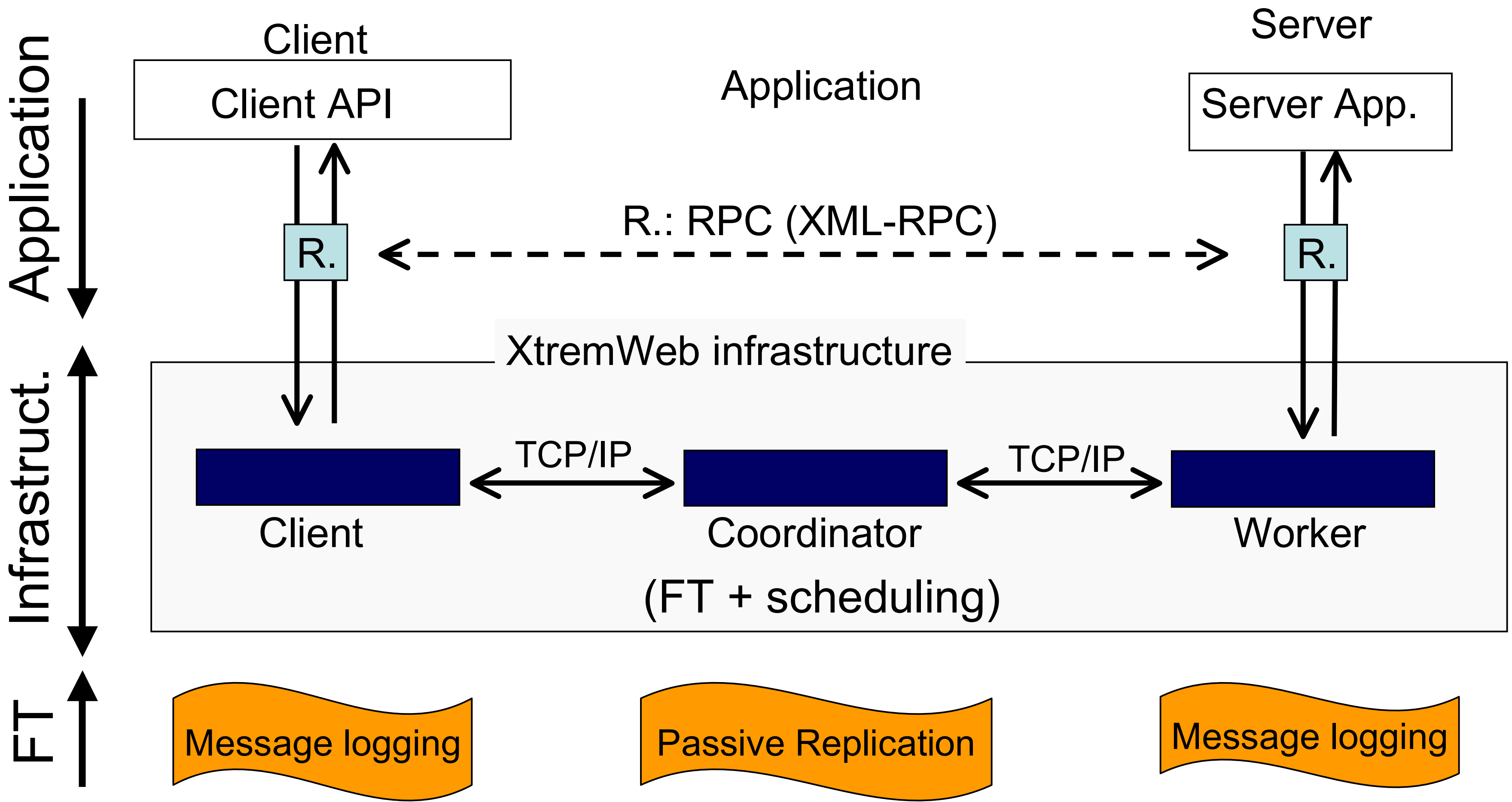
1) Automatic fault tolerance
2) Transparent for the programmer & user
3) Tolerate Client and Server faults
4) Firewall bypass
5) Avoid global synchronizations (ckpt/restart)
6) Theoretical verification of protocols

RPC-V Design

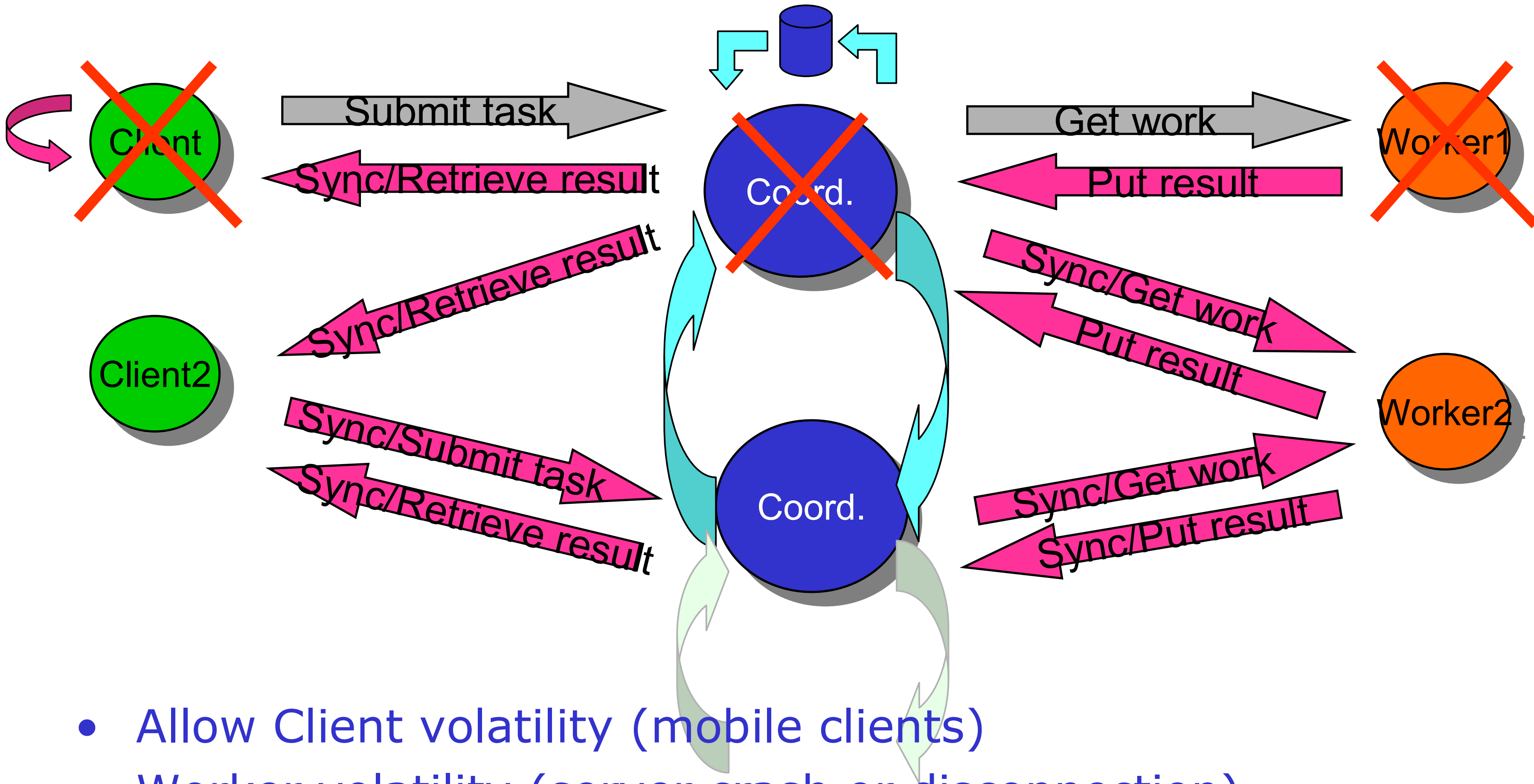
Asynchronous network (Internet + P2P volatility)?

If yes → restriction to stateless or single user statefull apps.

If no → muti-users statefull apps. (needs atomic broadcast)



RPC-V in Action

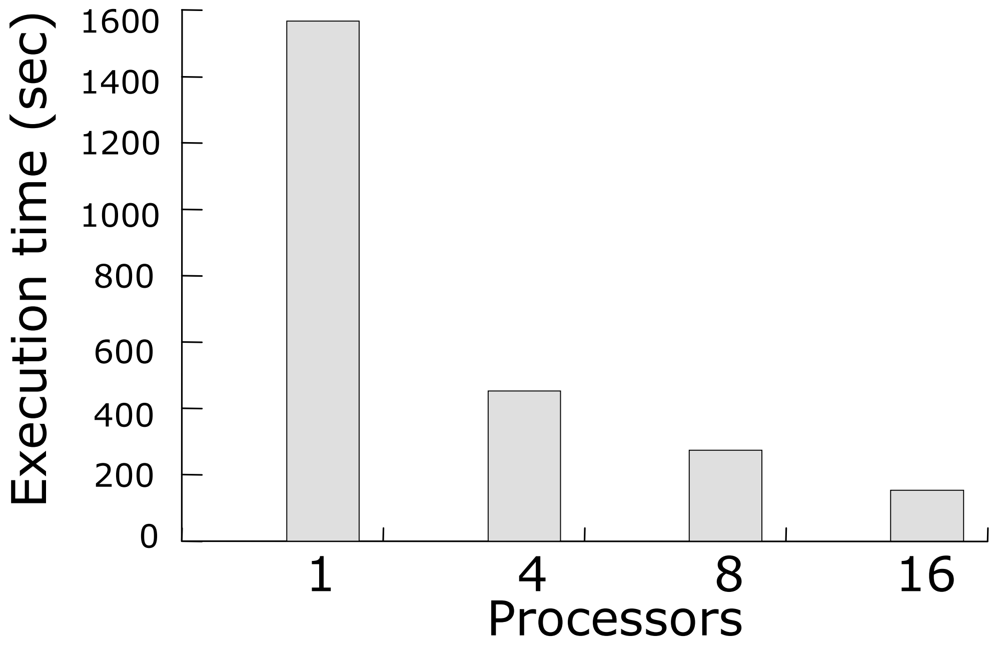


- Allow Client volatility (mobile clients)
- Worker volatility (server crash or disconnection)
- Coordinator crash or transient faults (warning: task may be executed more than once)

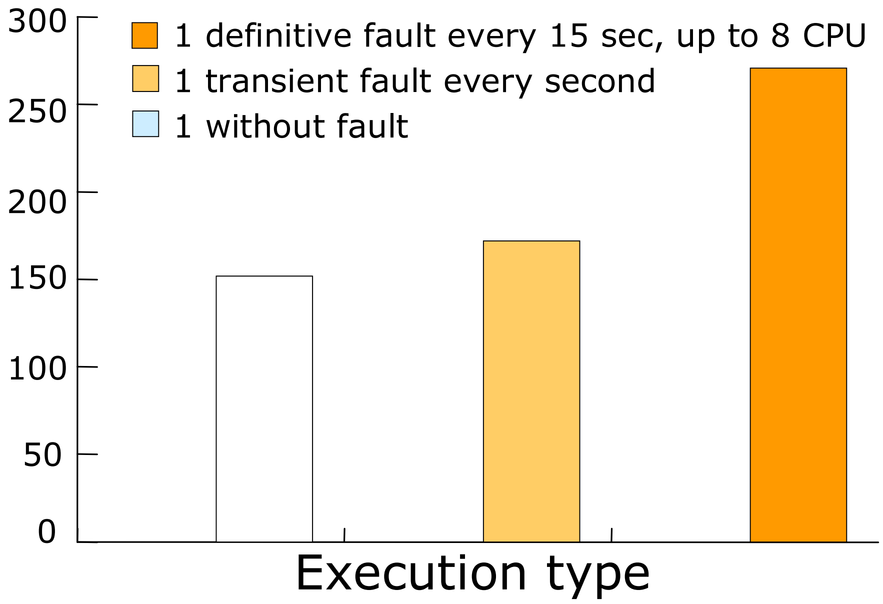
RPC-V performance

NAS EP Class C (16 nodes), Athlon 1800+ and 100BT Ethernet
100 tasks (15 sec. each)

Fault free execution



Execution with faults



- As long as the application is already available on server, transient fault have a very low impact on performance (10%)

MPICH-V (Volatile)

Goal: execute existing or new MPI Apps

Programmer’s view unchanged:

PC client MPI_send()

PC client MPI_recv()

Problems: 1) volatile nodes (any number at any time)
2) firewalls (PC Grids)
3) non named receptions (→ should be replayed in the same order as the one of the previous failed exec.)

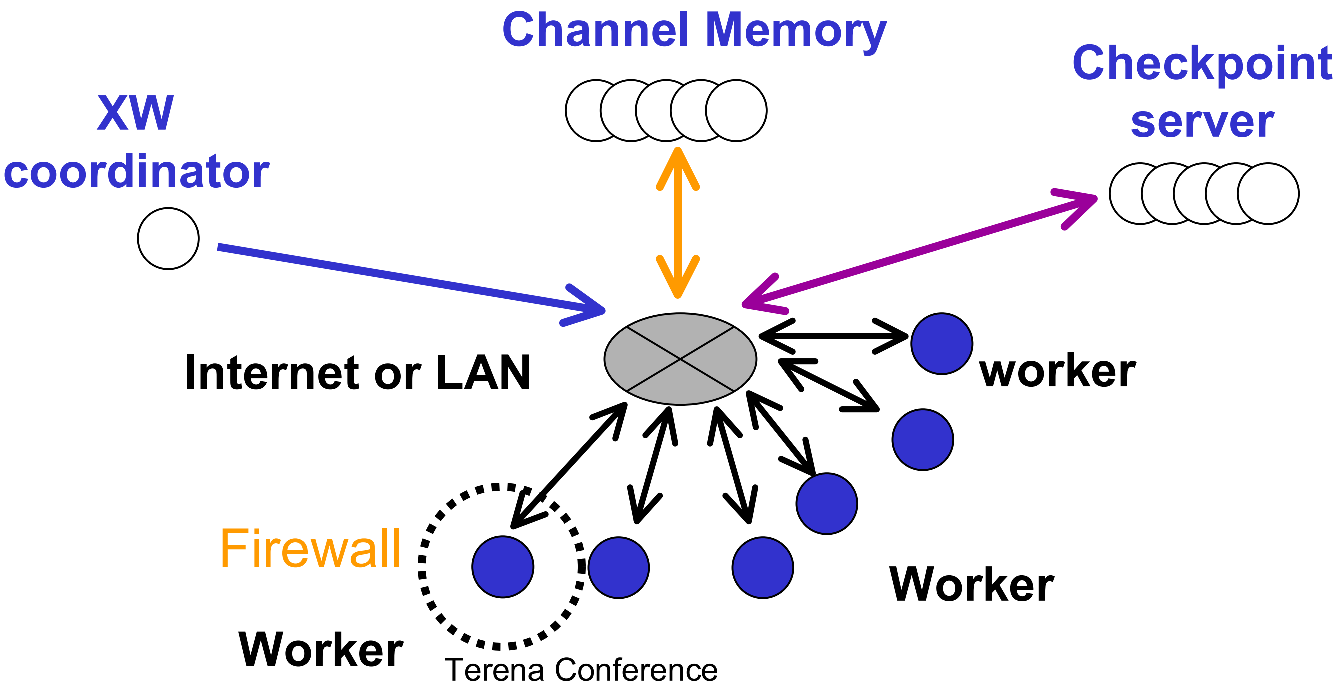
Objective summary:

1) Automatic fault tolerance
2) Transparency for the programmer & user
3) Tolerate n faults (n being the #MPI processes)
4) Firewall bypass (tunnel) for cross domain execution
5) Scalable Infrastructure/protocols
6) Avoid global synchronizations (ckpt/restart)
7) Theoretical verification of protocols

MPICH-V: Global architecture

MPICH-V :

- –Communications : a MPICH device with Channel Memory
- –Run-time : virtualization of MPICH processes in XW tasks with checkpoint
- –Linking the application with libxwmpi instead of libmpich



MPICH-V: Channel Memories

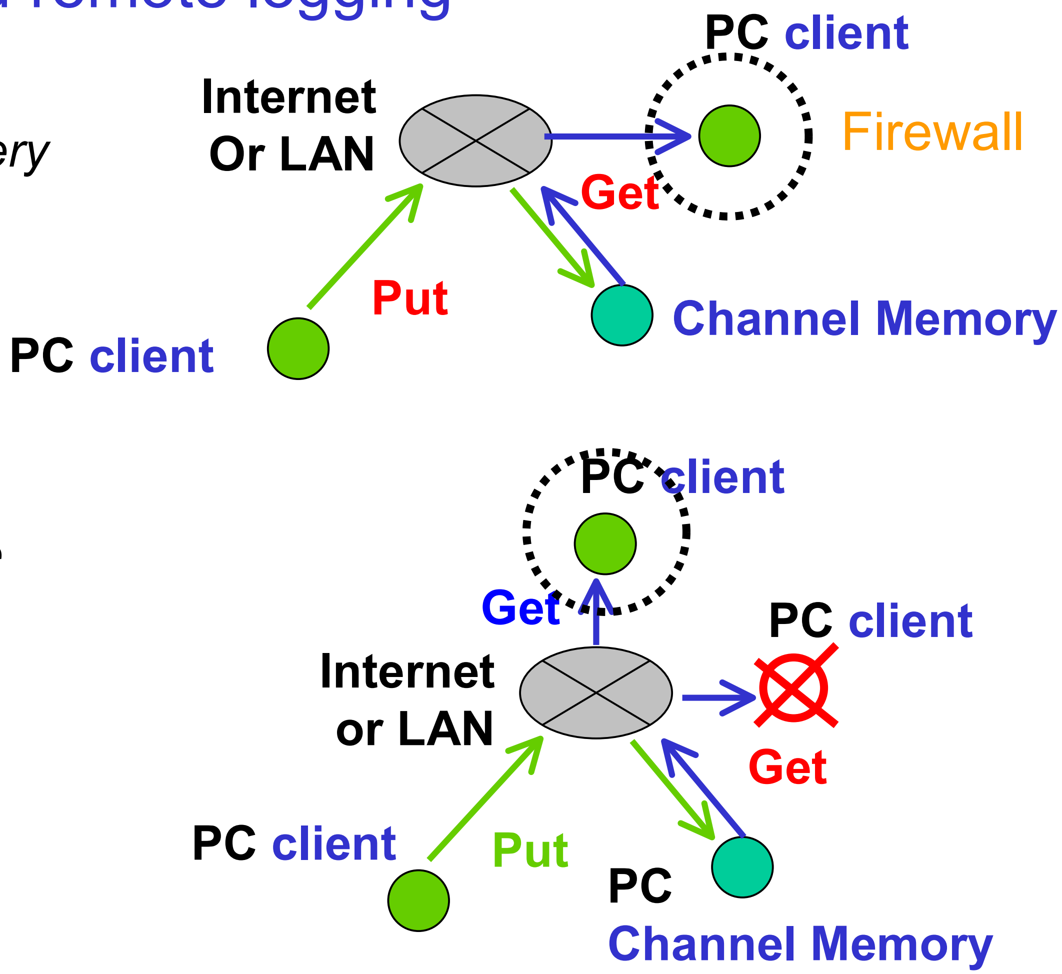
Pessimistic distributed remote logging

A set of reliable nodes called "Channel Memories" logs every message.

All communications are Implemented by 1 PUT and 1 GET operation to the CM

PUT and GET operations are transactions

When a process restarts, it replays all communications using the Channel Memory

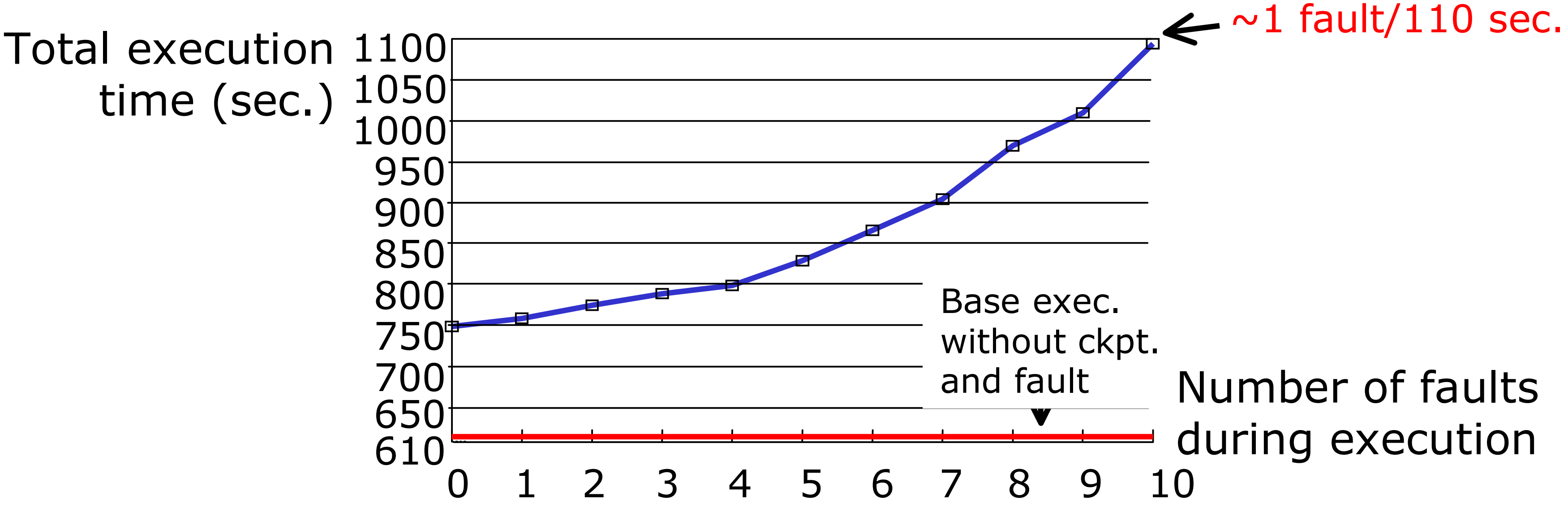


Advantage: no global restart; Drawback: performance

Performance with volatile nodes

Performance of BT.A.9 with frequent faults

- 3 CM, 2 CS (4 nodes on 1 CS, 5 on the other)
- 1 checkpoint every 130 seconds on each node (non sync.)



→ Overhead of ckpt is about 23%
→ For 10 faults performance is 68% of the one without fault

→ MPICH-V allows application to survive node volatility (1 F/2 min.)
→ Performance degradation with frequent faults stays reasonable

Outline

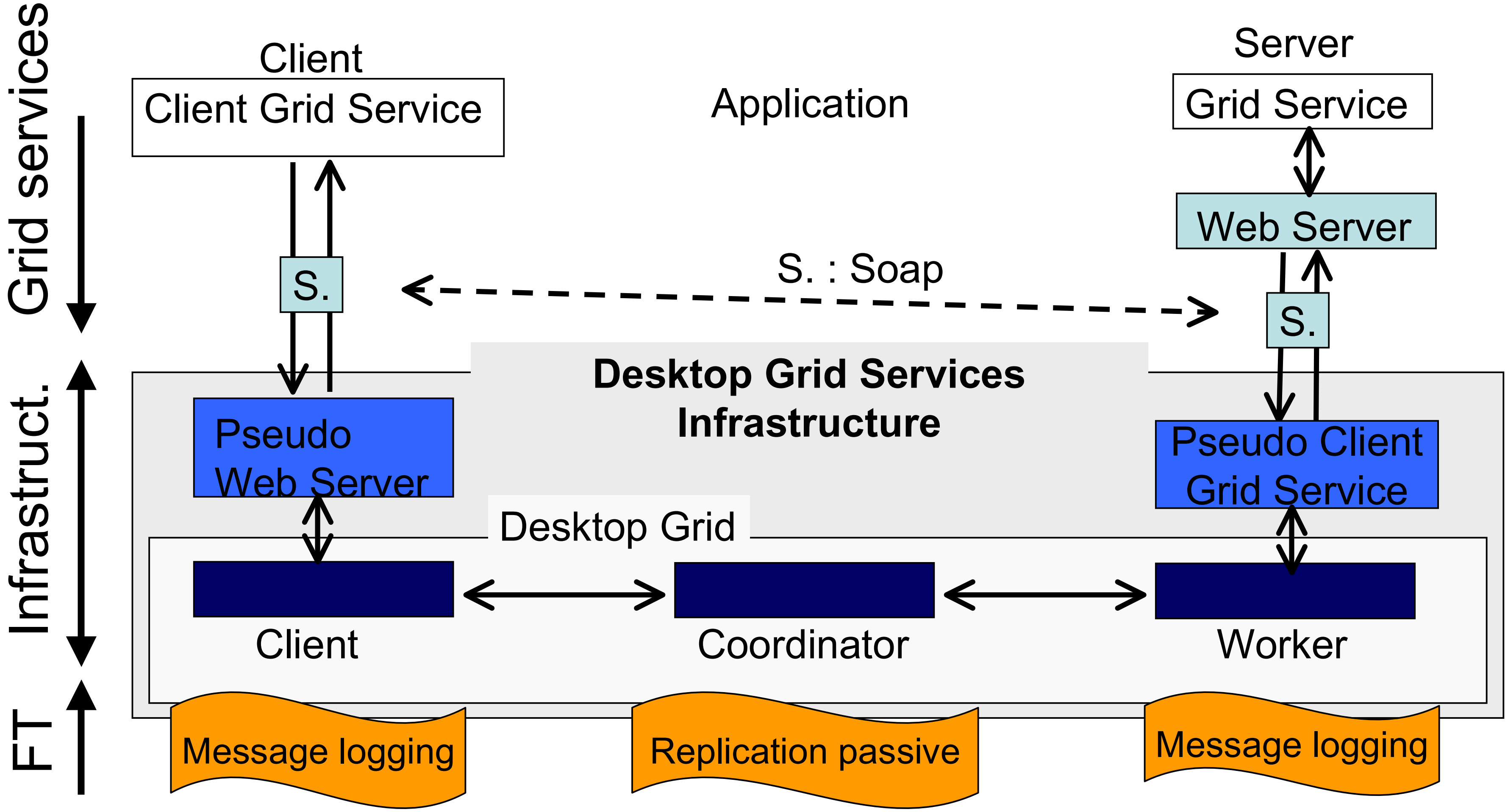
- Introduction (large scale distributed systems)
- System issues in HPC P2P infrastructure
  - Internal of P2P systems for computing
  - Case Studies: XtremWeb / BOINC
- Programming HPC P2P infrastructures
  - RPC-V
  - MPICH-V (A message passing library For XtremWeb)
- Open issue: merging Grid & P2P
- Concluding remarks

Grand Large

Merging Grid and P2P:

Executing Grid Services on P2P systems:

A variant of RPC-V: DGSI



Concluding Remark

High performance computing on P2P systems
is a long term effort:

Many issues are still to be solved:

- Global architecture (distributed coordination)
- User Interface, control language
- Security, sandboxing
- Large scale storage
- Message passing library (RPC-V, MPICH-V)
- Scheduling -large scale, multi users, muti app.-
- *GRID/P2P interoperability*
- Validation on real applications

Bibliography

[2] Projet ACI GRID CGP2P, www.lri.fr/~fci/CGP2P

[3] Projet XtremWeb, www.xtremweb.net

[4] Third « Global P2P Computing » Workshop coallocated with IEEE/ACM CCGRID 2003, Tokyo, Mai 2003, http://gp2pc.lri.fr

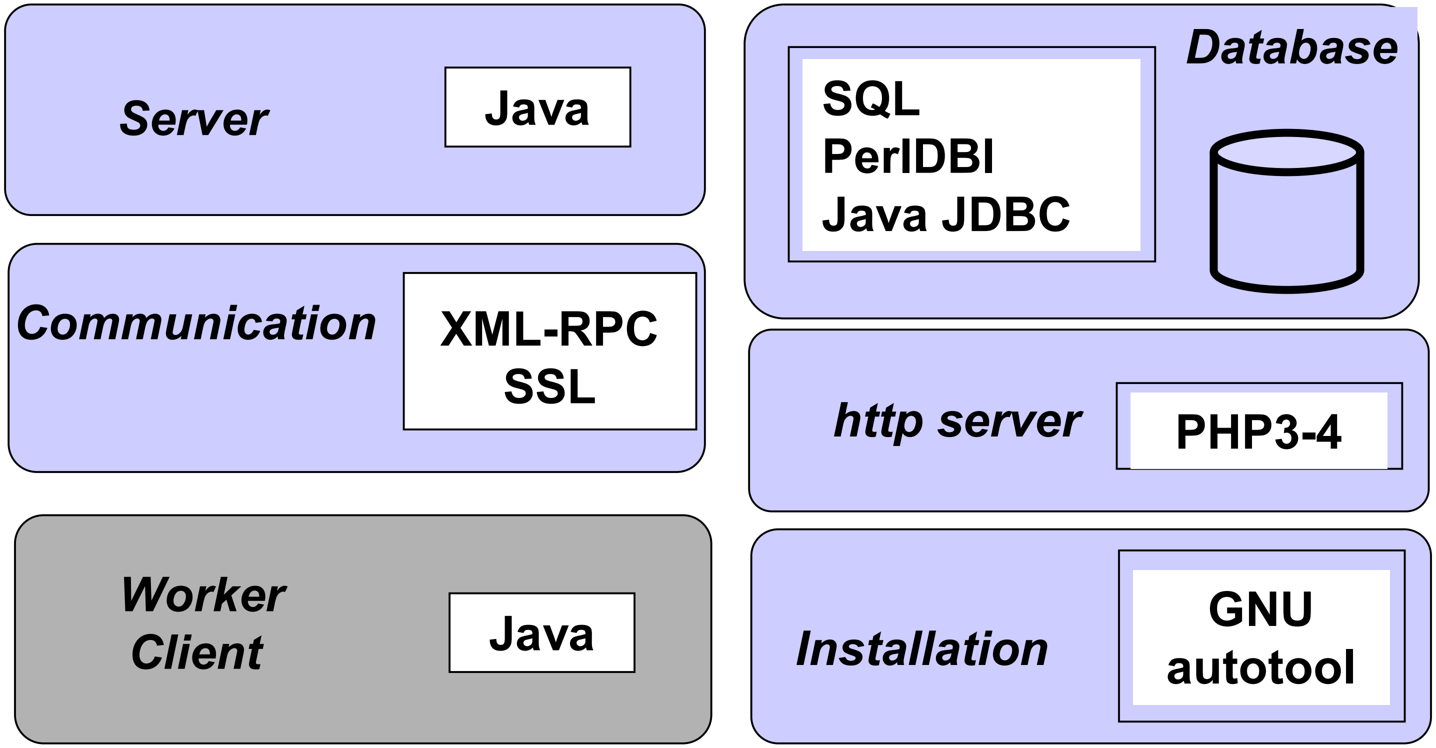
[5] « Peer-to-Peer Computing », D. Barkai, Intel press, 2001, Octobre 2001.

[6] « Harnessing the power of disruptive technologies”, A. Oram éditeur, edition O’Reilly, Mars 2001

[7] « Search in power-law networks », L. A. Adamic et al. Physical Review, E Volume 64, 2001

[8] « The Grid : Blueprint for a new Computing Infrastructure », I. Foster et C. Kesselman, Morgan-Kaufmann, 1998.

XtremWeb Software Technologies



Server: Java

Communication: XML-RPC SSL

Worker Client: Java

Database: SQL, PerlDBI, Java JDBC

http server: PHP3-4

Installation: GNU autotool

Installation prerequisites : database (Mysql), web server (apache), PHP, JAVA jdk1.2.

XtremWeb recent developments

Installation

- Easier installation with Apache Ant (a sort of make)

Architecture

- Stand alone Workers (can be launched using a Batch Scheduler) - a single jar file.
- Coordinator API (used for replication, scheduling, etc.)

Programming models

- Fault tolerant RCP (called RPC-V)
- RPC-V + Grid Services = DGSI (Desktop Grid Services Infrastructure)
- MPICH-V2 (second version of MPICH-V)
- C-MPICH (Checkpointable MPICH)

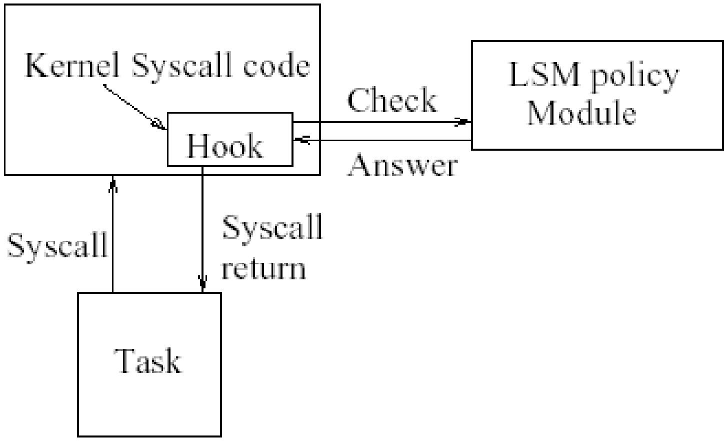
Effort on Scheduling: fully distributed

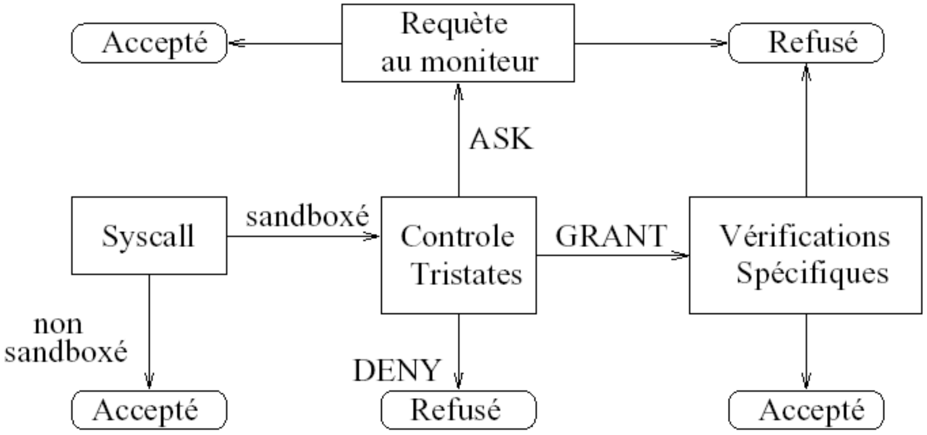
- New algorithms (Sand heap, Hot potatoes, Score tables)
- Methodology: Theoretical, Swarm (High level simulator), MicroGrid (Emulator), XtremWeb (Testbed)

Security : SBSLM
Frederic Magniette (Post Doc ACI)

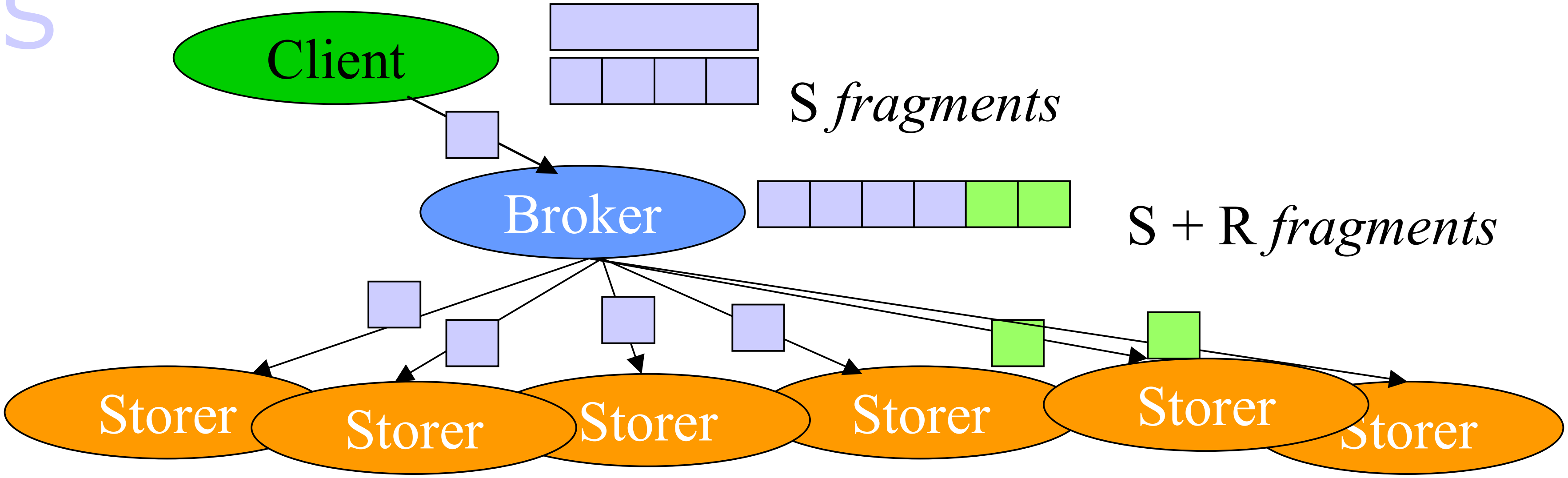
Sandbox module based on LSM (kernel programming in C).
Principle : a user level security policy for a set of sandboxed processes

For each security hook, SBLSM starts checking a dedicated variable (set by the user) concerning this hook which may take three states:
- GRANT, specific verifications are executed.
- DENY, access denied returning the –EACCES error code.
- ASK, user request via the security device.





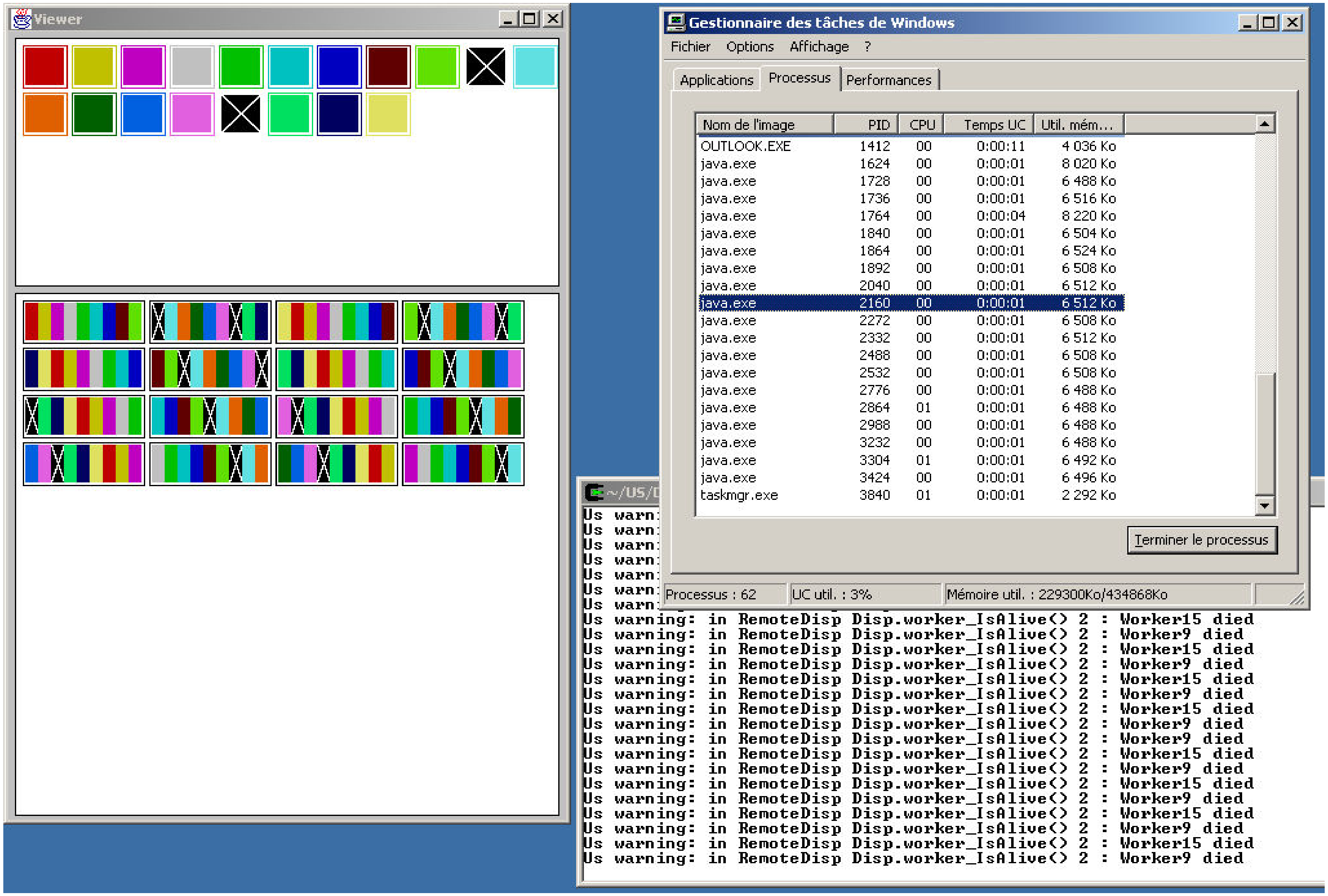
Storage on P2P systems: US (LARIA/LIP)



- Brocker
  - new ()
  - malloc (taille) ➔ Space
- Space
  - put (index, buffer)
  - get (index) ➔ buffer
  - free (index)

```
Brocker brocker = new Brocker (193.10.32.01);
Space space = brocker.malloc(1000);
…
    for (i=0; i<100; i++) {
      buffer = fileIn.read (space.getBlockSize());
      space.put (i, buffer);
    }
    …
    for (i=0; i<100; i++) {
      buffer = space.get (i);
      fileOut.write (buffer, space.getBlockSize);
    }
```

Storage on P2P systems: US



Nom de l'image	PID	CPU	Temps UC	Util. mém...
OUTLOOK.EXE	1412	00	0:00:11	4 036 Ko
java.exe	1624	00	0:00:01	8 020 Ko
java.exe	1728	00	0:00:01	6 488 Ko
java.exe	1736	00	0:00:01	6 516 Ko
java.exe	1764	00	0:00:04	8 220 Ko
java.exe	1840	00	0:00:01	6 504 Ko
java.exe	1864	00	0:00:01	6 524 Ko
java.exe	1892	00	0:00:01	6 508 Ko
java.exe	2040	00	0:00:01	6 512 Ko
java.exe	2160	00	0:00:01	6 512 Ko
java.exe	2272	00	0:00:01	6 508 Ko
java.exe	2332	00	0:00:01	6 512 Ko
java.exe	2488	00	0:00:01	6 508 Ko
java.exe	2532	00	0:00:01	6 508 Ko
java.exe	2776	00	0:00:01	6 488 Ko
java.exe	2864	01	0:00:01	6 488 Ko
java.exe	2988	00	0:00:01	6 488 Ko
java.exe	3232	00	0:00:01	6 488 Ko
java.exe	3304	01	0:00:01	6 492 Ko
java.exe	3424	00	0:00:01	6 496 Ko
taskmgr.exe	3840	01	0:00:01	2 292 Ko

Processus : 62 UC util. : 3% Mémoire util. : 229300Ko/434868Ko

```
Us warn:
Us warn:
Us warn:
Us warn:
Us warn:
Us warn:
Us warn:
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker15 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker9 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker15 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker9 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker15 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker9 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker15 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker9 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker9 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker15 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker9 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker15 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker9 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker15 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker9 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker15 died
Us warning: in RemoteDisp Disp.worker_IsAlive() 2 : Worker9 died
```