A Client-Server Approach for Simulation over the Grid

Frédéric Desprez
LIP ENS Lyon
INRIA Rhône-Alpes
GRAAL Research Team

Join work with

A. Su, R. Bolze, E. Caron,
H. Dail, J.-Y. L'Excellent
LIP, Lyon

P. Amestoy, M. Pantel, C. Puglisi
ENSEEIHT/IRIT, Toulouse

CoreGRID · Ministère de la Recherche · ACI GRID · CNRS Centre National de la Recherche Scientifique · INRIA · Grid'5000

Introduction

- One simple (and efficient) paradigm for grid computing
  - Offering (or renting) computational power and/or storage capacity through the Internet
  - Providing access to existing applications to thin clients

☺ Very high potential

- Need of Problem Solving and Application Service Provider Environments
- Installation difficulty for some libraries and applications
- Some libraries or codes need to stay where they have been developed
- Some data need to stay in place for security reasons

→ Using computational servers through a simple interface

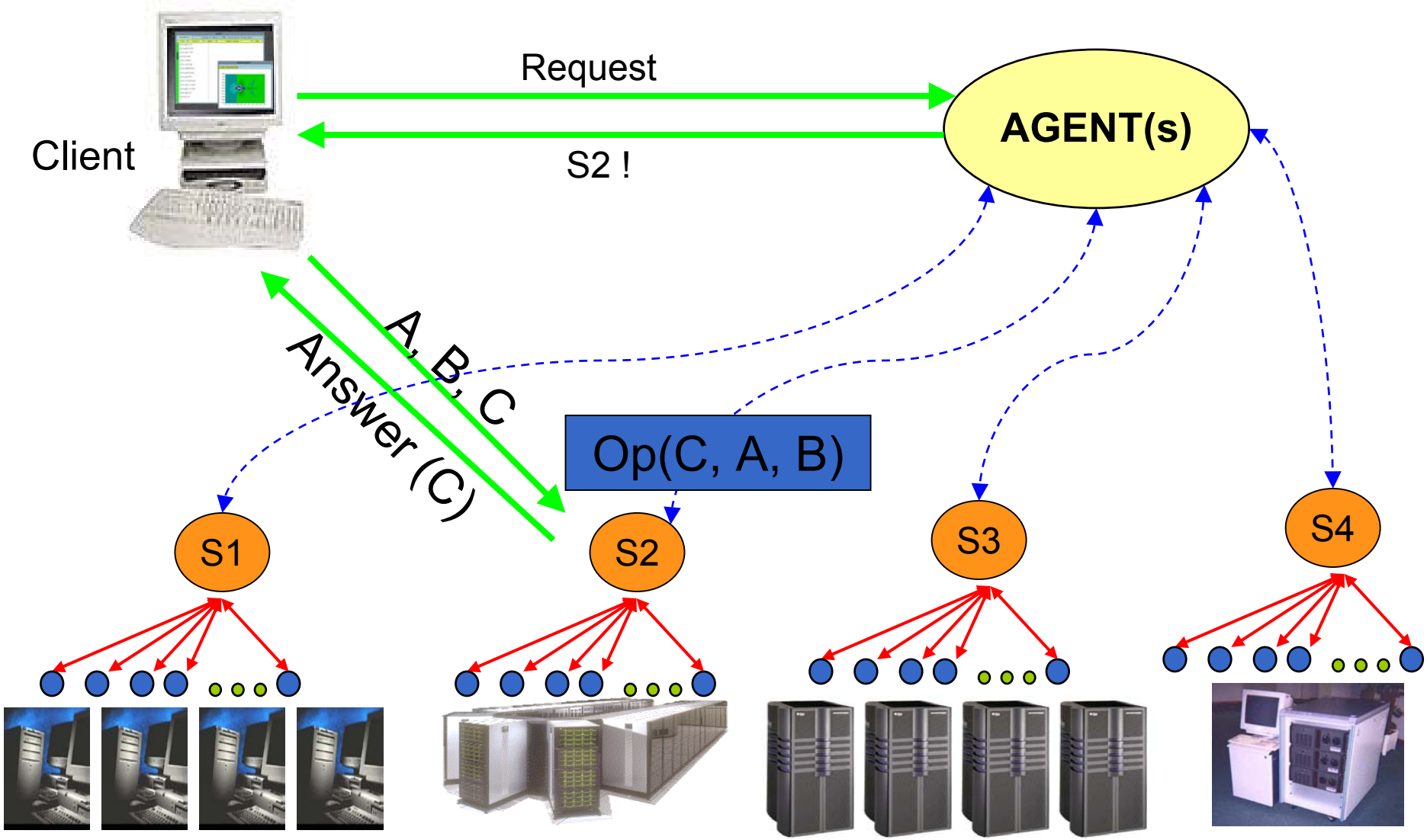
☹ But

- Always hard to use for non-specialists
- Often application dependent PSEs
- No sophisticated scheduling

RPC and Grid-Computing: GridRPC

- **One simple idea**
  - Implementing the RPC programming model over the grid
  - Using resources accessible through the network
  - Mixed parallelism model (data-parallel model at the server level and task parallelism between the servers)
- **Features needed**
  - Load-balancing (resource localization and performance evaluation, scheduling),
  - IDL,
  - Data and replica management,
  - Security,
  - Fault-tolerance,
  - Interoperability with other systems,
  - …
- **Design of a standard interface**
  - within the GGF (GridRPC WG, C. Lee)
  - www.ggf.org, forge.gridforum.org/projects/gridrpc-wg
  - Existing implementations: NetSolve, Ninf, DIET, XtremWeb

RPC and Grid Computing: Grid RPC



RPC and Grid Computing: Grid RPC

- Adaptable grain
- Simple RPC API
- Libraries and applications integrated in Grid components
- IDL for the client interface, minimal information
- Task parallelism at the client/server level (using asynchronous calls), Data-parallelism at the server level $\Rightarrow$ mixed parallelism

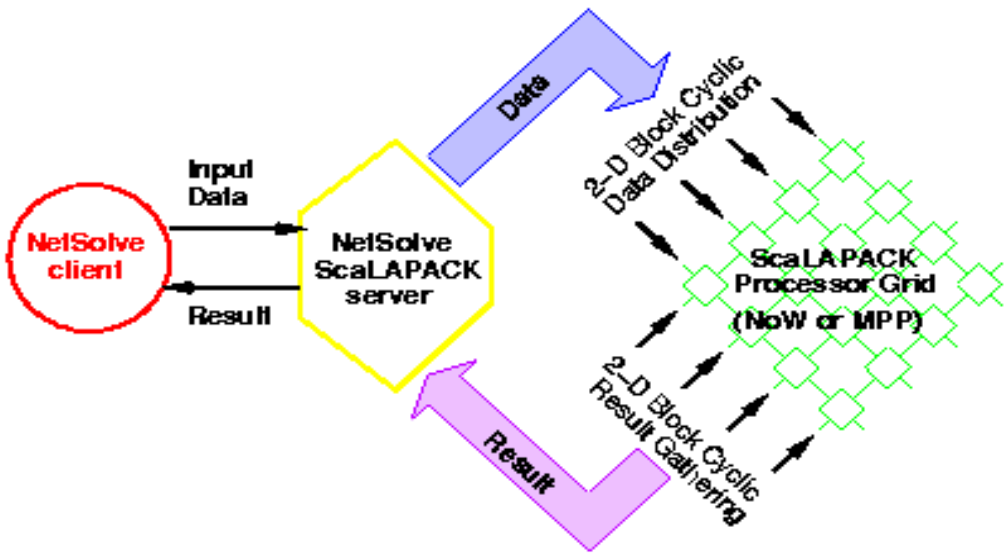
```
double A[n][n],B[n][n],C[n][n];          /* data declaration  */
dmmul(n,A,B,C);                          /* local function call */

GRPC_call("dmmul",n,A,B,C);              /* remote function call */
```

Hidden parallelism to the user

- One sequential call in the client code
  - Data transfer to the target server (maybe parallel one)
  - Resource reservation on the server
  - Distribution for the target parallel routine chosen by the server(/agent)
  - Execution of the parallel code on the server (with or without check-pointing for fault-tolerance)
  - Gathering of the result and send to the client (pipeline?)

- Transparent for the client code !



RPC and Grid Computing: Grid RPC

Five fundamental components:

- Client

  Offers several user's interface and submit requests to servers

- Server

  Receive clients requests and executes software modules on behalf of them

- Data-base

  Contains both static and dynamic information about hardware and software resources

- Scheduler

  Gets clients requests and takes decisions to map tasks on servers depending of data stored in the database

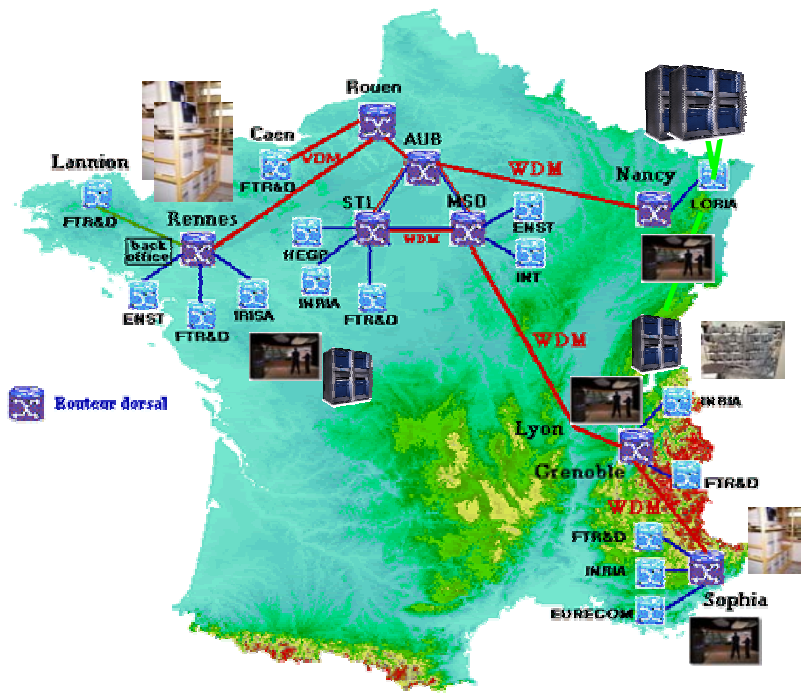
- Monitor

  makes observations about resources status and stores information in the database

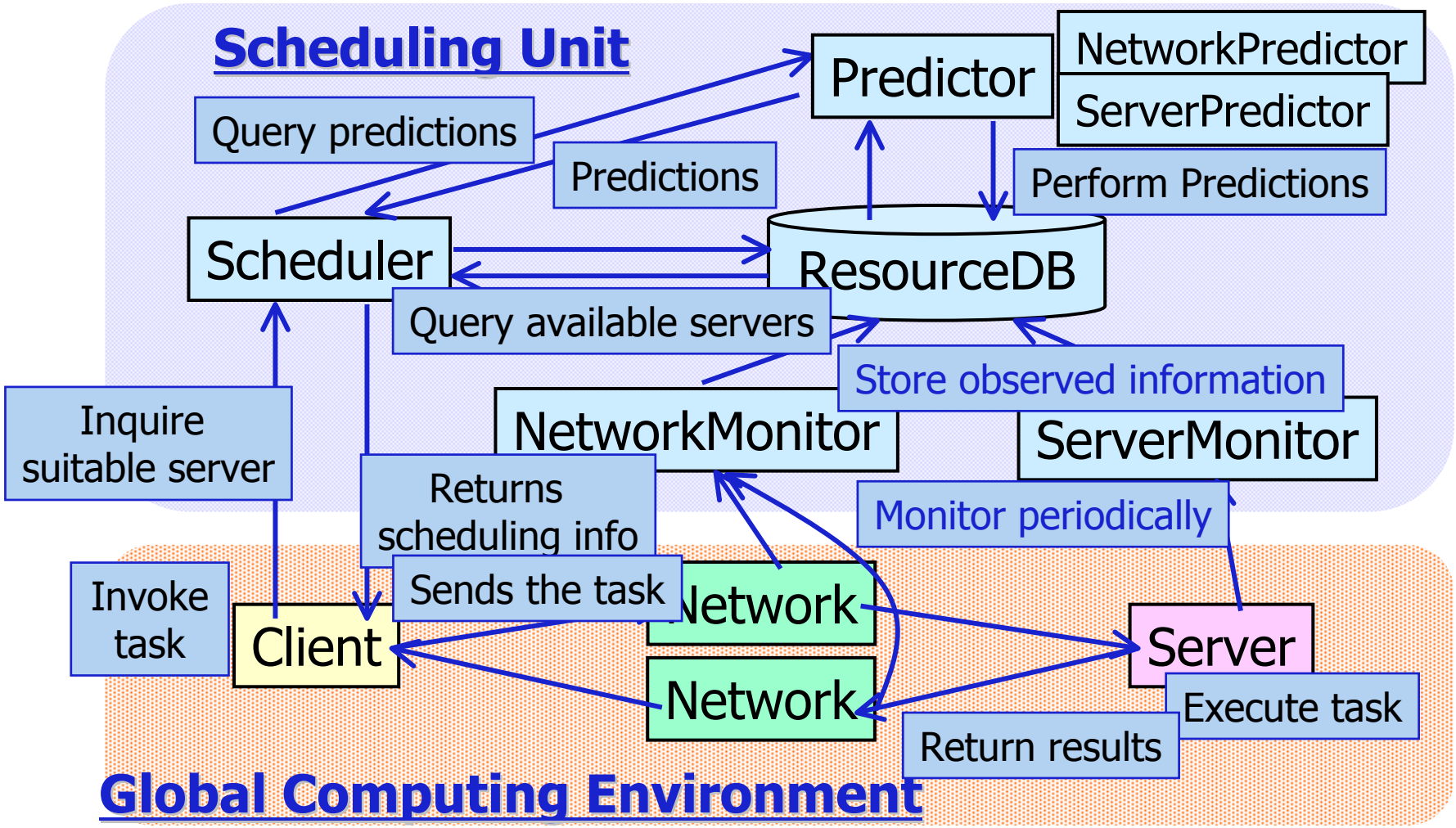
AGENT

- Central component of GRID-RPC systems
- Choose servers able to solve a request on behalf of clients
- Main task: load-balancing between servers
  - Gets information about available servers
  - Asks the performance database for information
  - Applies some scheduling heuristics
  - Can take care of
    - Some security (access autorization)
    - Fault tolerance
- ‘Smart’ localization mandatory
- Some scalability problems may occur

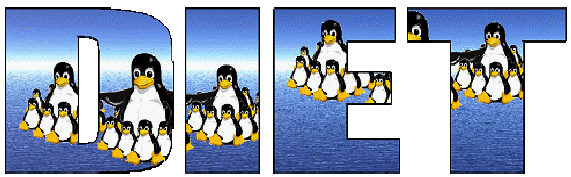
- Centralized (or duplicated) in NetSolve or Ninf
- Distributed in DIET



Agent Behavior



Distributed Interactive Engineering Toolbox

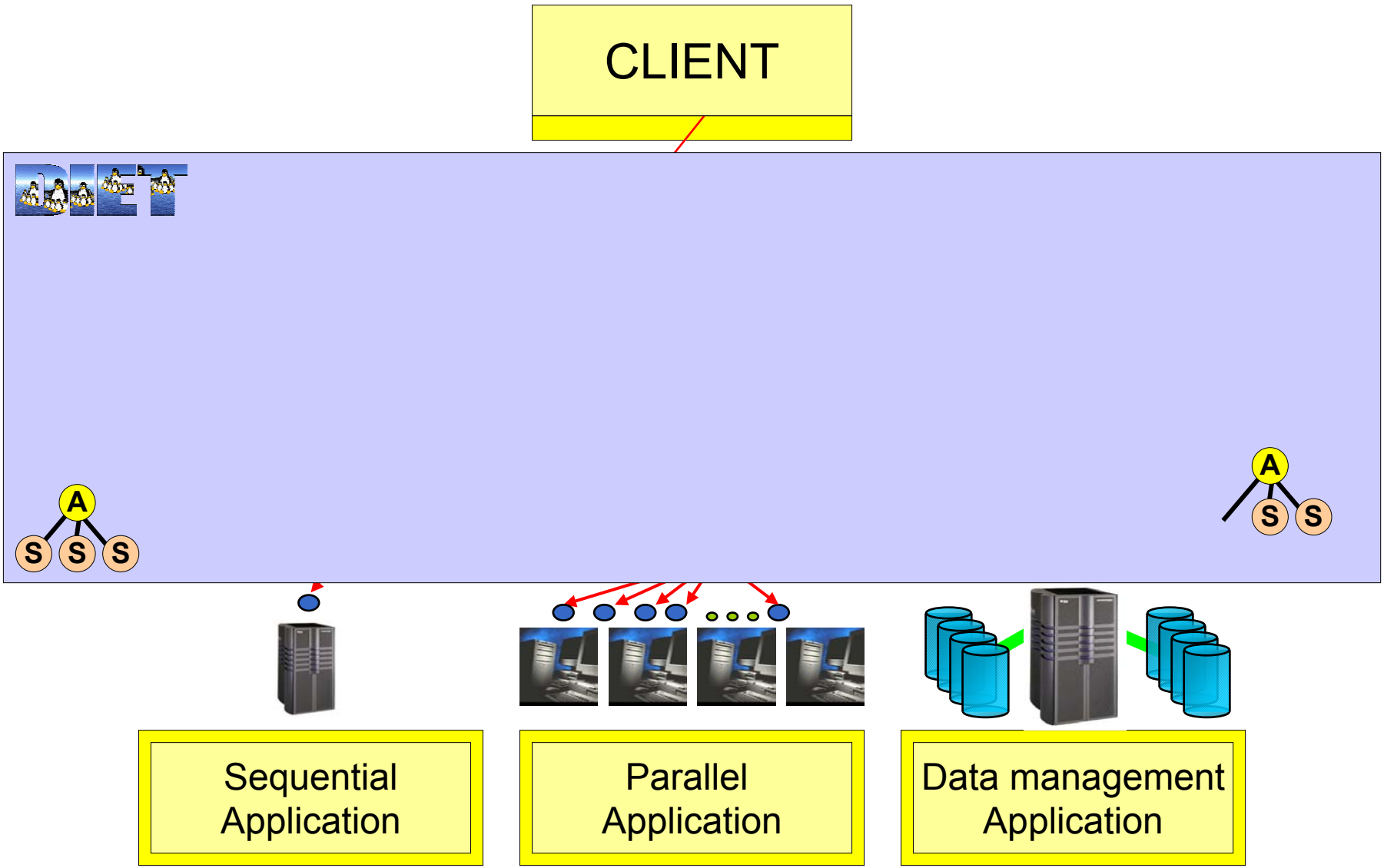


DIET's Goals

- Our goals
  - To develop a toolbox for the deployment of environments using the Application Service Provider (ASP) paradigm with different applications
  - Use as much as possible public domain and standard software
  - To obtain a high performance and scalable environment
  - Implement and validate our more theoretical results
    - Scheduling for heterogeneous platforms, data (re)distribution and replication, performance evaluation, algorithmic for heterogeneous and distributed platforms, …
- Based on CORBA, NWS, LDAP, and our own software developments
  - FAST for performance evaluation,
  - LogMgr for monitoring,
  - VizDIET for the visualization,
  - GoDIET for the deployment
- Several applications in different fields (simulation, bioinformatic, …)
- Release 1.1 available on the web
- ACI Grid ASP, RNTL GASP

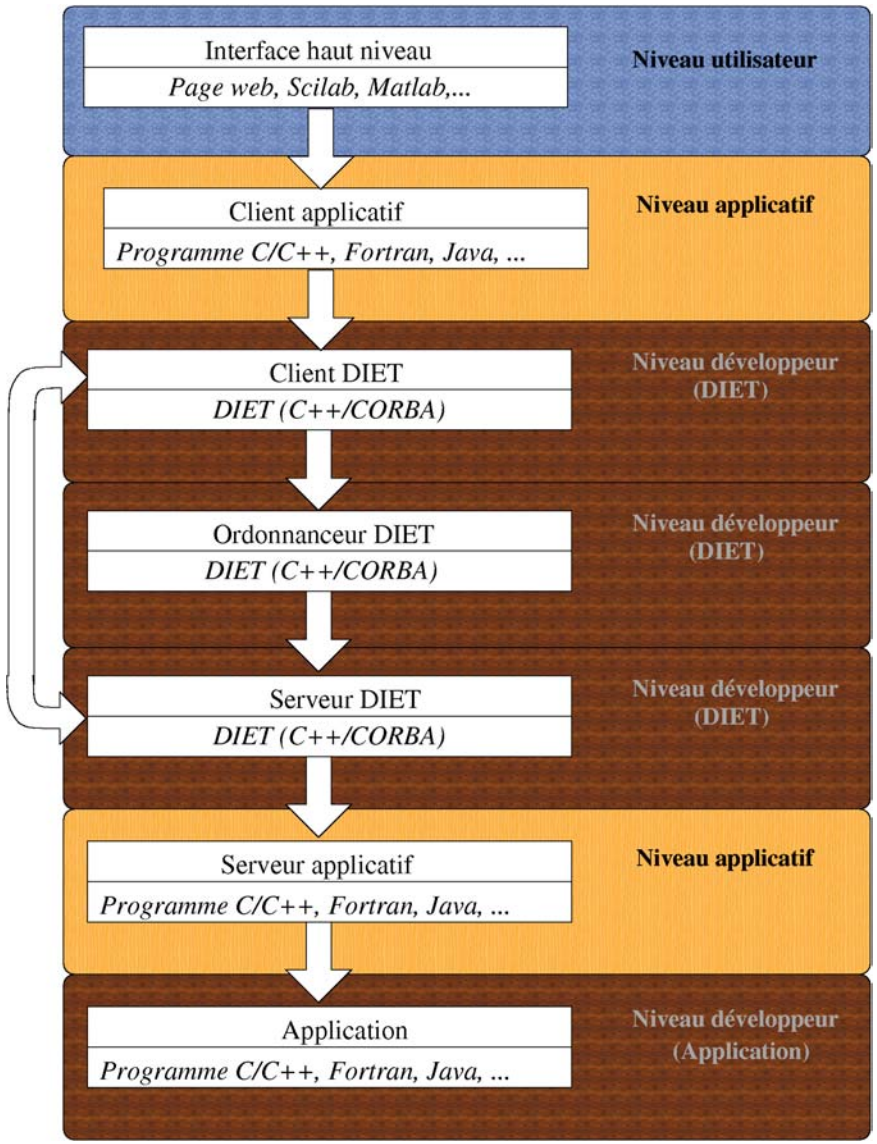
http://graal.ens-lyon.fr/DIET/

DIET Environment

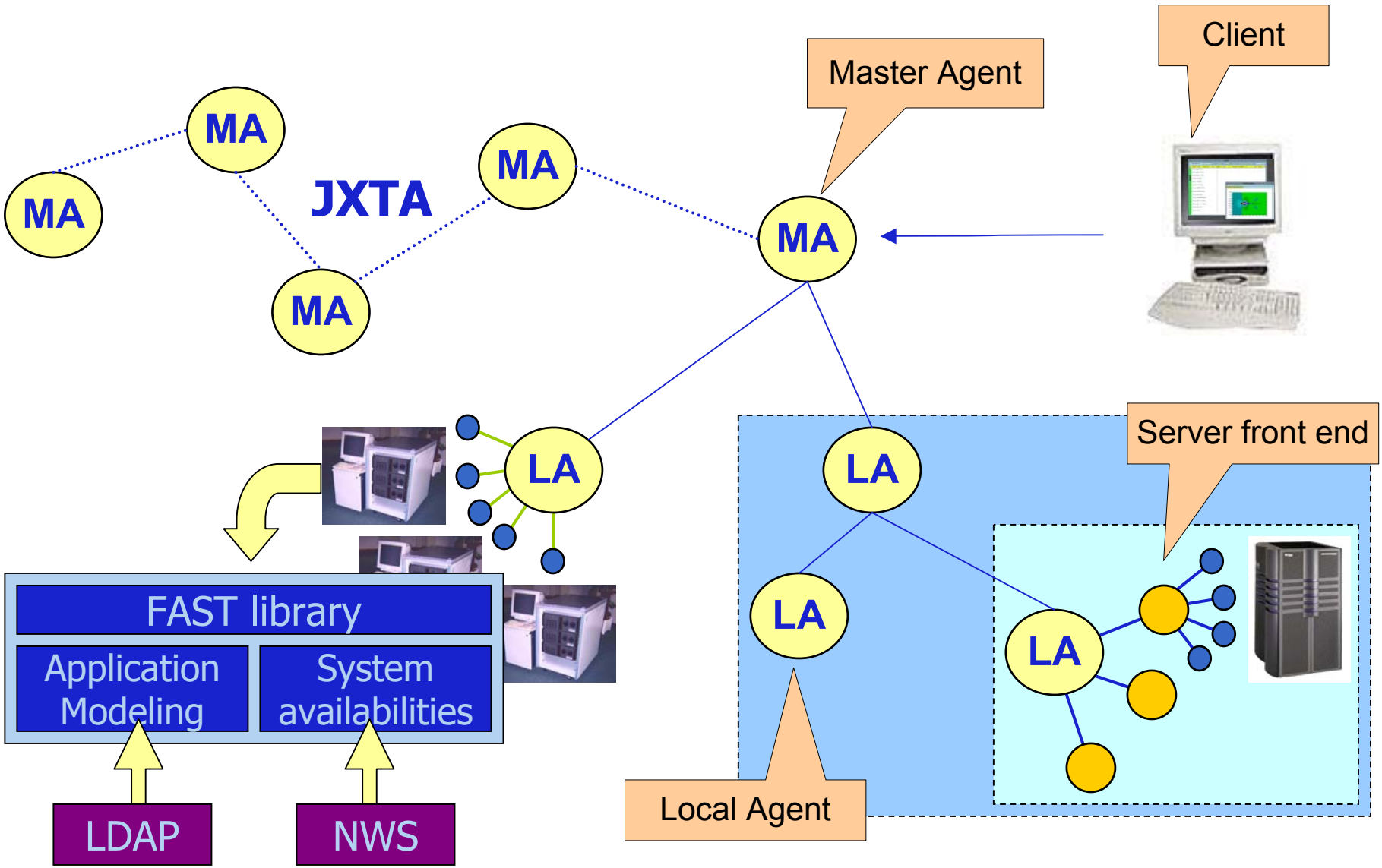


Client Interface

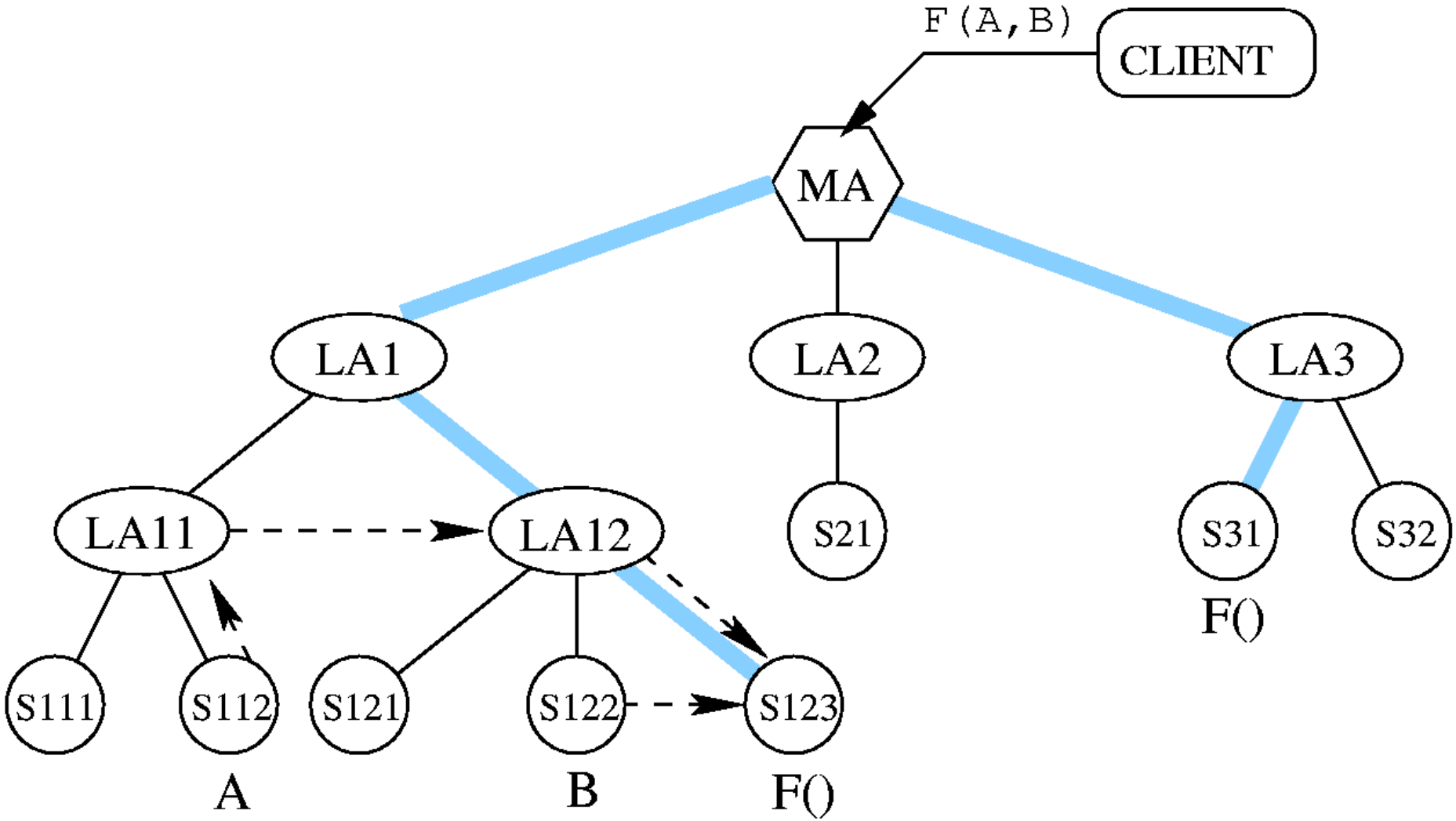
- As simple as possible
- Multi-interfaces (C, C++, Fortran, Java, Matlab, Mathematica, Scilab, Web, ...)
- Proposition of a standard interface within the Global Grid Forum (DIET, Ninf, and Netsolve)



DIET Architecture



Request Management



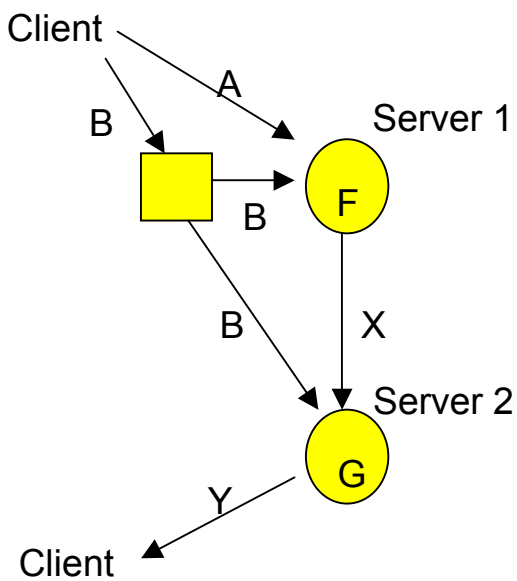
Some Research Topics

- Scheduling
  - Distributed scheduling
  - Software platform deployment with or without dynamic connections between components
  - Plug-in schedulers
- Data-management
  - Scheduling of computation requests and links with data-management
  - Replication, data prefetching
  - Workflow scheduling
- Performance evaluation
  - Application modelization
  - Dynamic information about the platform (network, clusters)
- Applications
  - Bioinformatic, geology, physic, chemical engineering, sparse solvers evaluation, …

Data Management

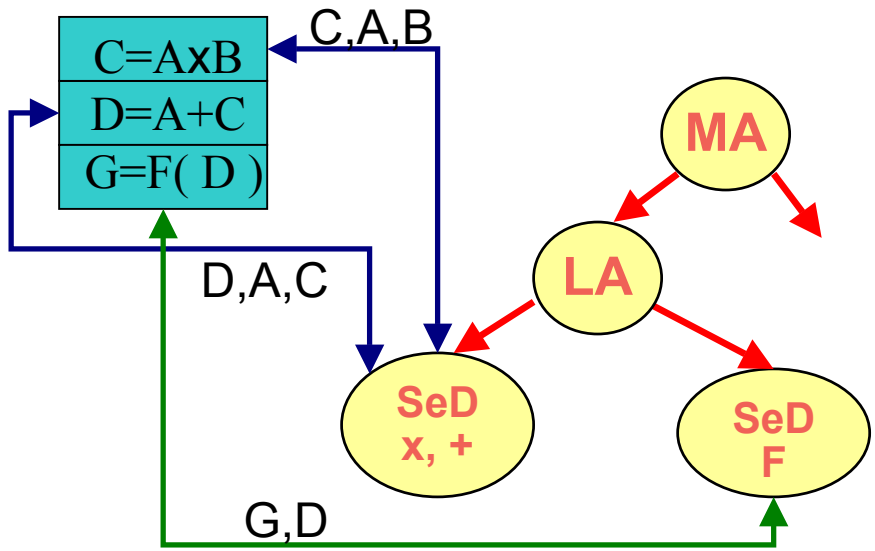
Data/replica management

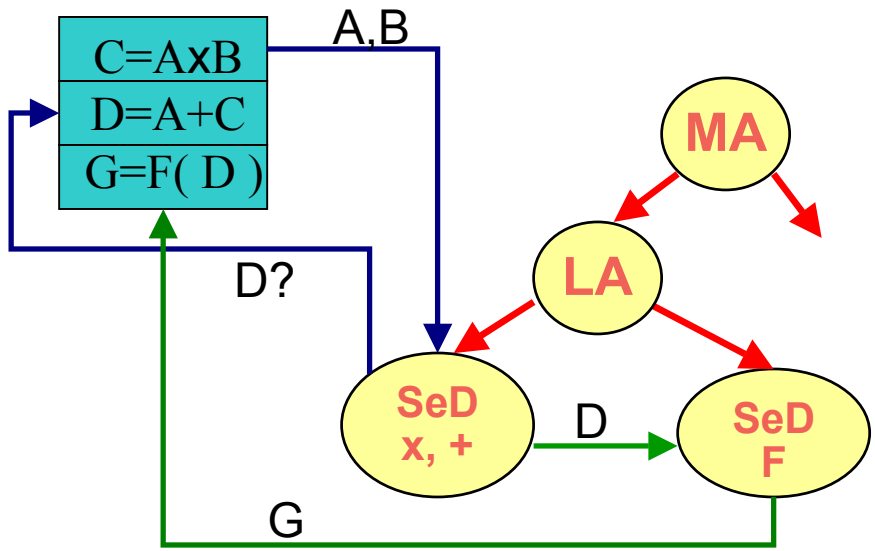
- Two needs
  - Keep the data in place to reduce the overhead of communications between clients and servers
  - Replicate data whenever possible
- Two approaches for DIET
  - DTM (LIFC, Besançon)
    - Hierarchy similar to the DIET's one
    - Distributed data manager
    - Redistribution between servers
  - JuxMem (Paris, Rennes)
    - P2P data cache
- NetSolve
  - IBP (Internet Backplane Protocol) : data cache
  - Request Sequencing to find data dependences
- Work done within the GridRPC Working Group
  - Relations with workflow management



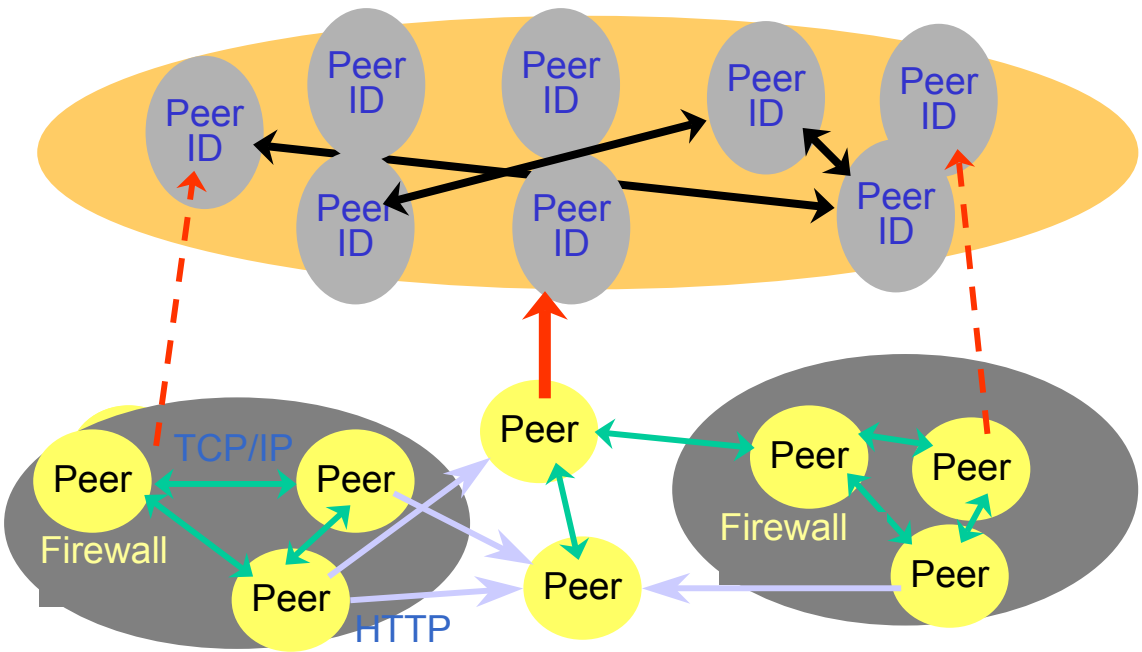
Data management with DTM within DIET

- Persistence at the server level
- To avoid useless data transfers
  - Intermediate results (C, D)
  - Between clients and servers
  - Between servers
  - “transparent” for the client
- Data Manager/Loc Manager
  - Hierarchy mapped on the DIET hierarchy
  - modularity
- Proposition to the Grid-RPC WG (GGF)
  - Data handles
  - Persistence flag
  - Data management functions





- A peer-to-peer architecture for a data-sharing service in memory
- Persistence and data coherency mechanism
- Transparent data localization

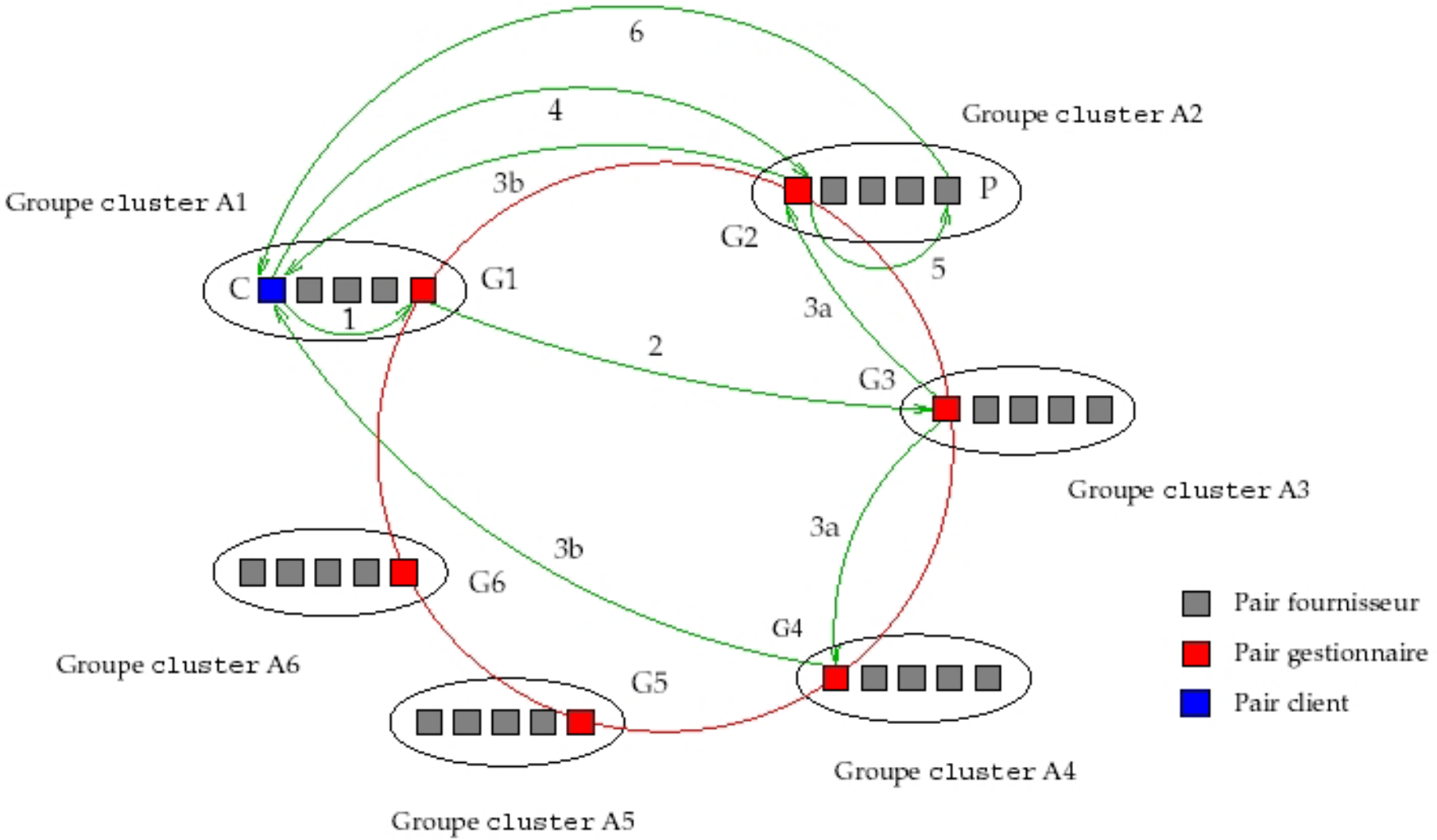


Project JXTA

- Toolbox for the development of P2P applications
  - Set of protocols
- One peer
  - Unique ID
  - Several communication protocols (TCP, HTTP, …)

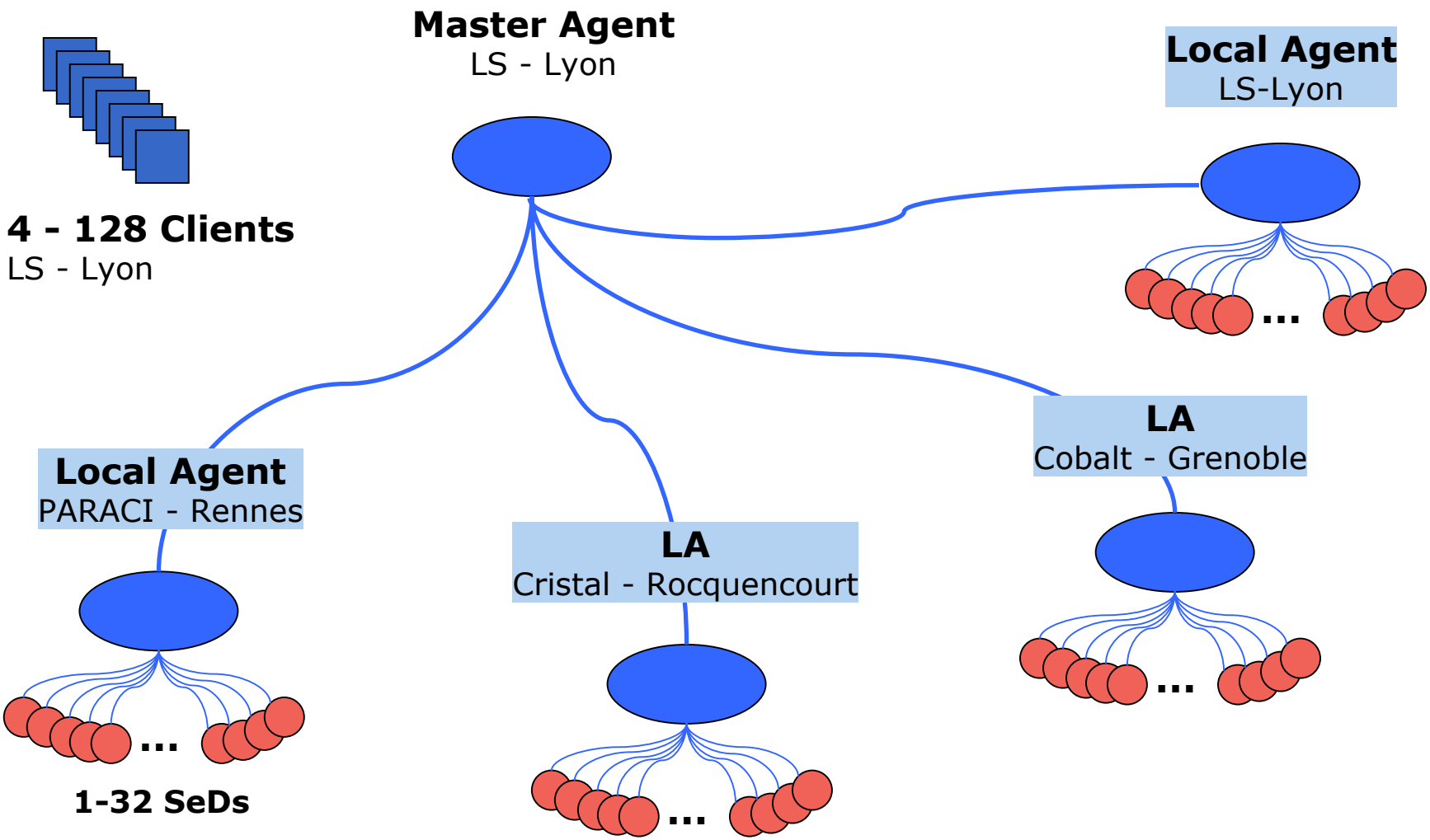
JuxMEM Architecture

- A peer-to-peer architecture for data-sharing

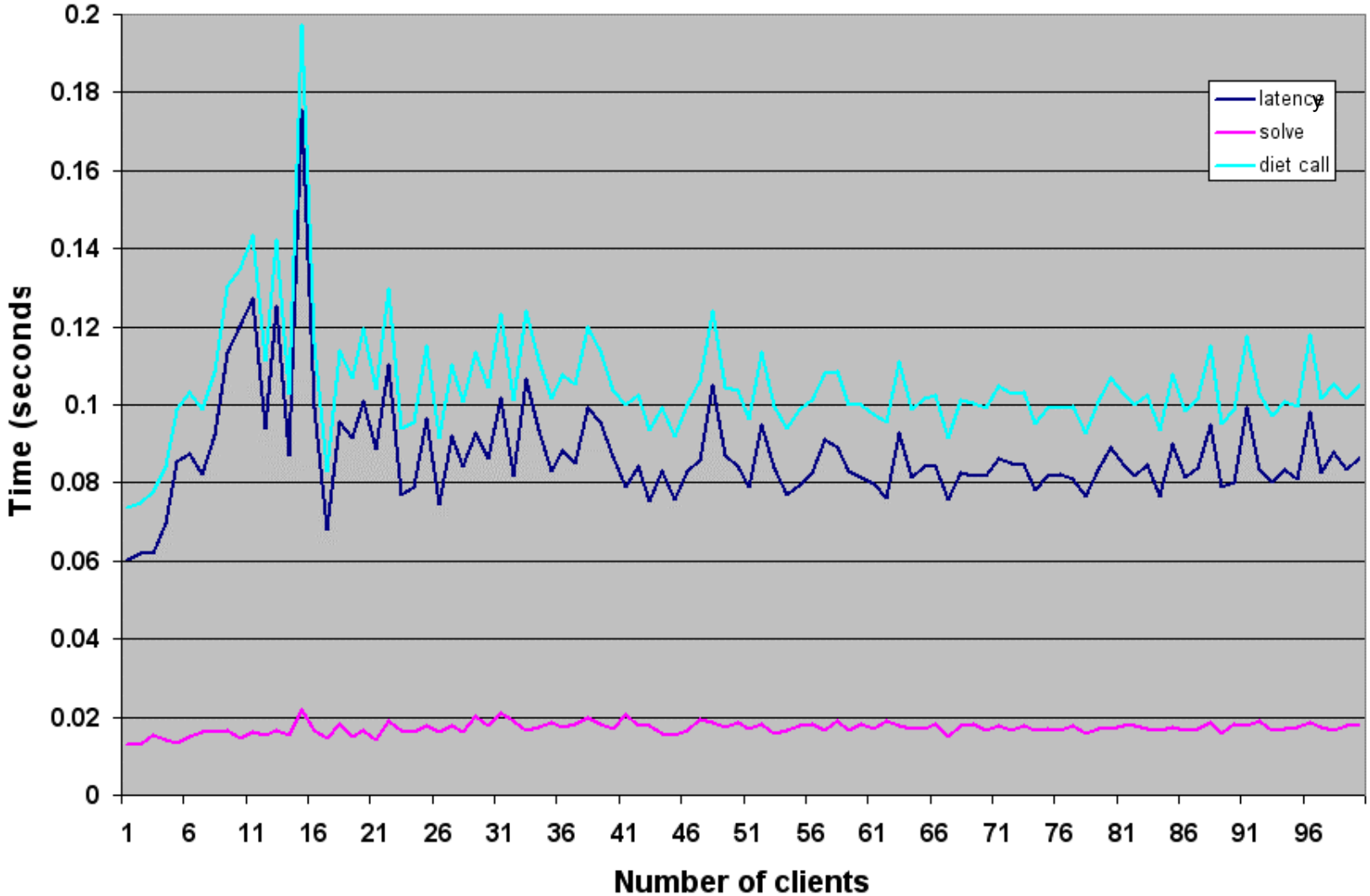


Experimentations

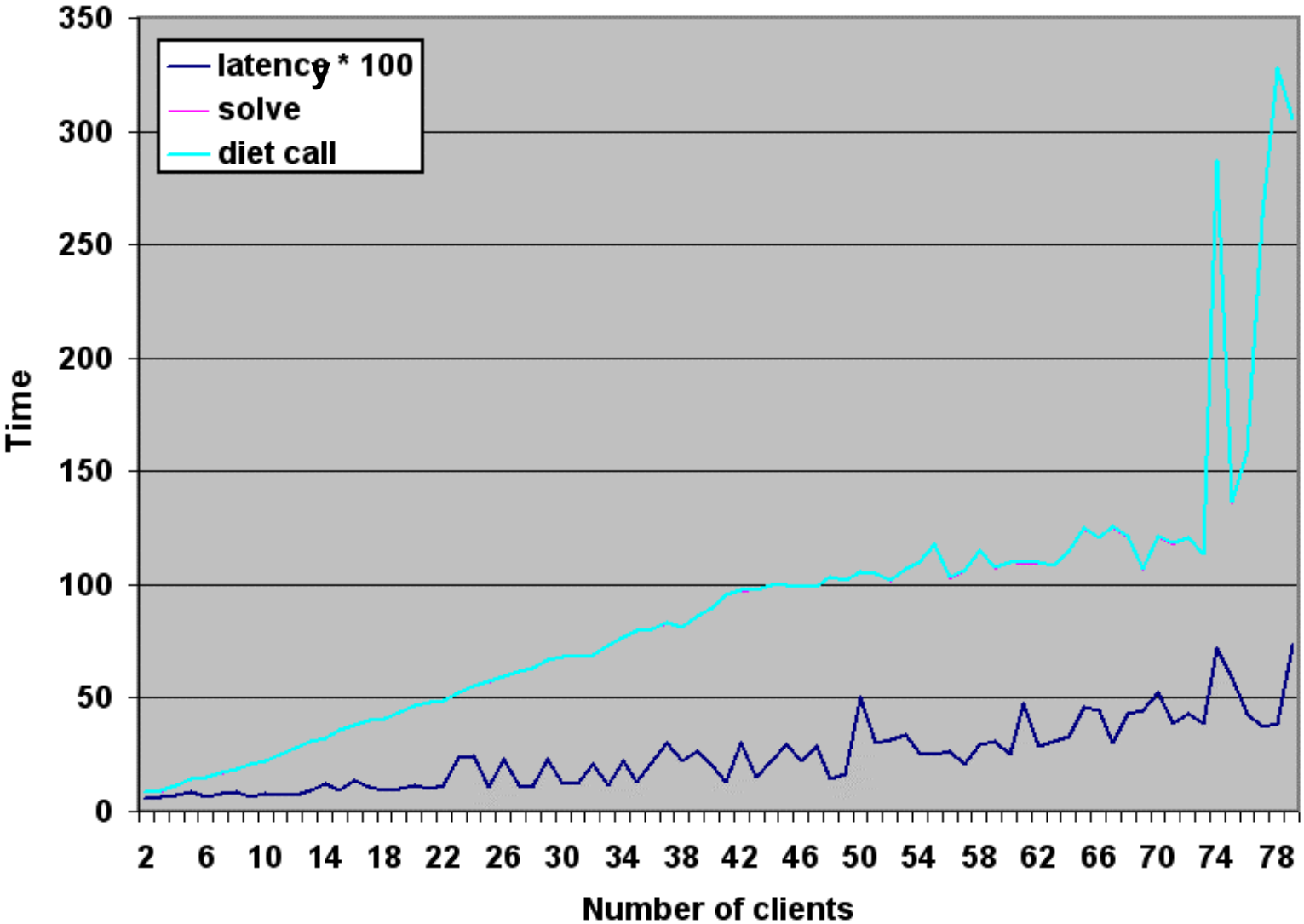
Target Platform



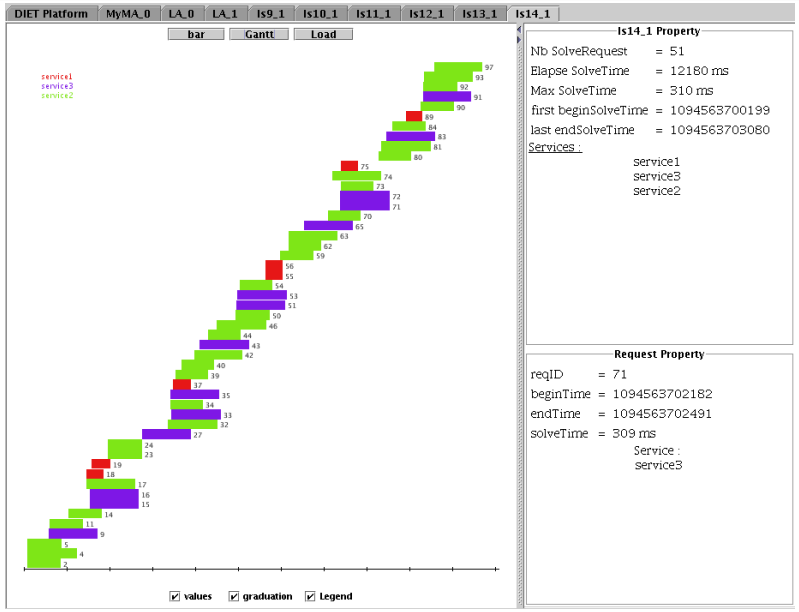
DIET Scalability with # clients. Size = 10.

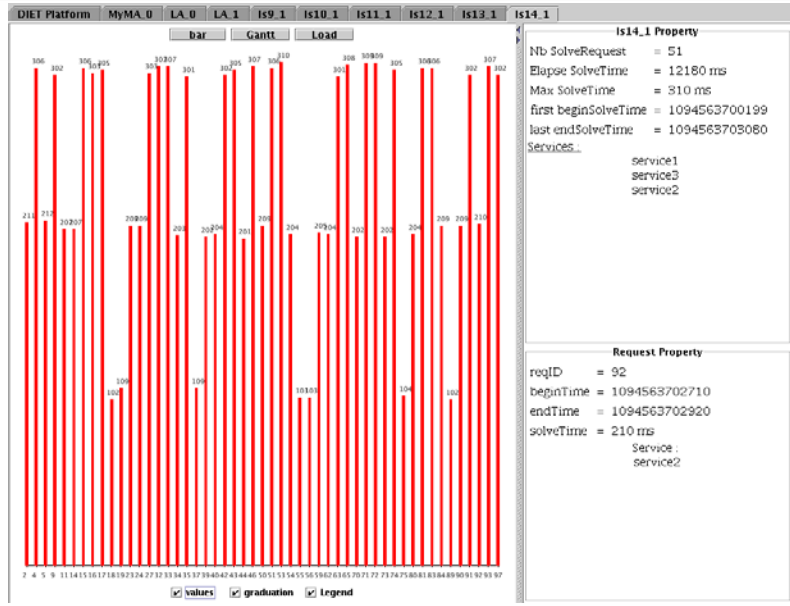


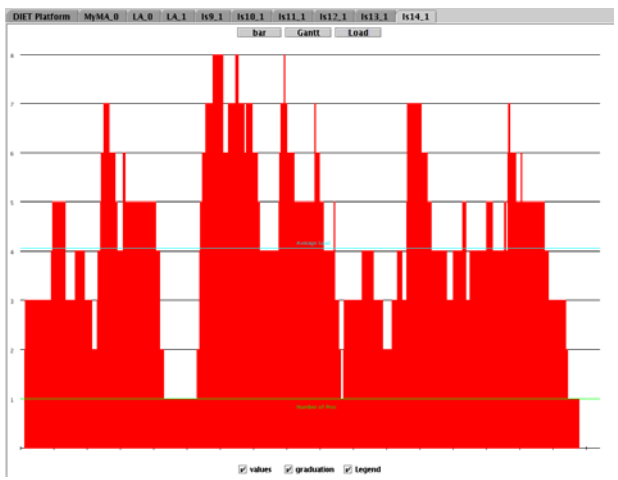
DIET Scalability with # clients. Size = 1000

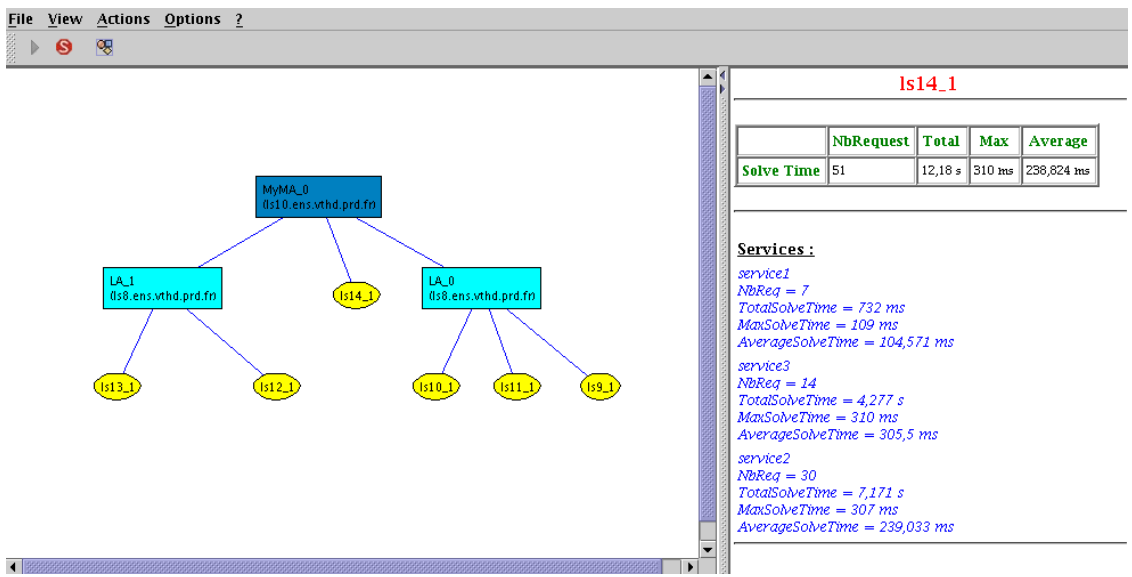


VizDIET


DIET Platform
MyMA_0
LA_0
LA_1
ls9_1
ls10_1
ls11_1
ls12_1
ls13_1
ls14_1
bar
Gantt
Load
ls14_1 Property
Nb SolveRequest = 51
Elapse SolveTime = 12180 ms
Max SolveTime = 310 ms
first beginSolveTime = 1094563700199
last endSolveTime = 1094563703080
Services :
service1
service3
service2
Request Property
reqID = 71
beginTime = 1094563702182
endTime = 1094563702491
solveTime = 309 ms
Service :
service3
values
graduation
Legend


DIET Platform
MyMA_0
LA_0
LA_1
ls9_1
ls10_1
ls11_1
ls12_1
ls13_1
ls14_1
bar
Gantt
Load
ls14_1 Property
Nb SolveRequest = 51
Elapse SolveTime = 12180 ms
Max SolveTime = 310 ms
first beginSolveTime = 1094563700199
last endSolveTime = 1094563703080
Services :
service1
service3
service2
Request Property
reqID = 92
beginTime = 1094563702710
endTime = 1094563702920
solveTime = 210 ms
Service :
service2
values
graduation
Legend


DIET Platform
MyMA_0
LA_0
LA_1
ls9_1
ls10_1
ls11_1
ls12_1
ls13_1
ls14_1
bar
Gantt
Load
values
graduation
Legend


File
View
Actions
Options
?
MyMA_0
(ls10.ens.vthd.prd.fr)
LA_1
(ls8.ens.vthd.prd.fr)
ls14_1
LA_0
(ls8.ens.vthd.prd.fr)
ls13_1
ls12_1
ls10_1
ls11_1
ls9_1
ls14_1
NbRequest
Total
Max
Average
Solve Time
51
12,18 s
310 ms
238,824 ms
Services :
service1
NbReq = 7
TotalSolveTime = 732 ms
MaxSolveTime = 109 ms
AverageSolveTime = 104,571 ms
service3
NbReq = 14
TotalSolveTime = 4,277 s
MaxSolveTime = 310 ms
AverageSolveTime = 305,5 ms
service2
NbReq = 30
TotalSolveTime = 7,171 s
MaxSolveTime = 307 ms
AverageSolveTime = 239,033 ms

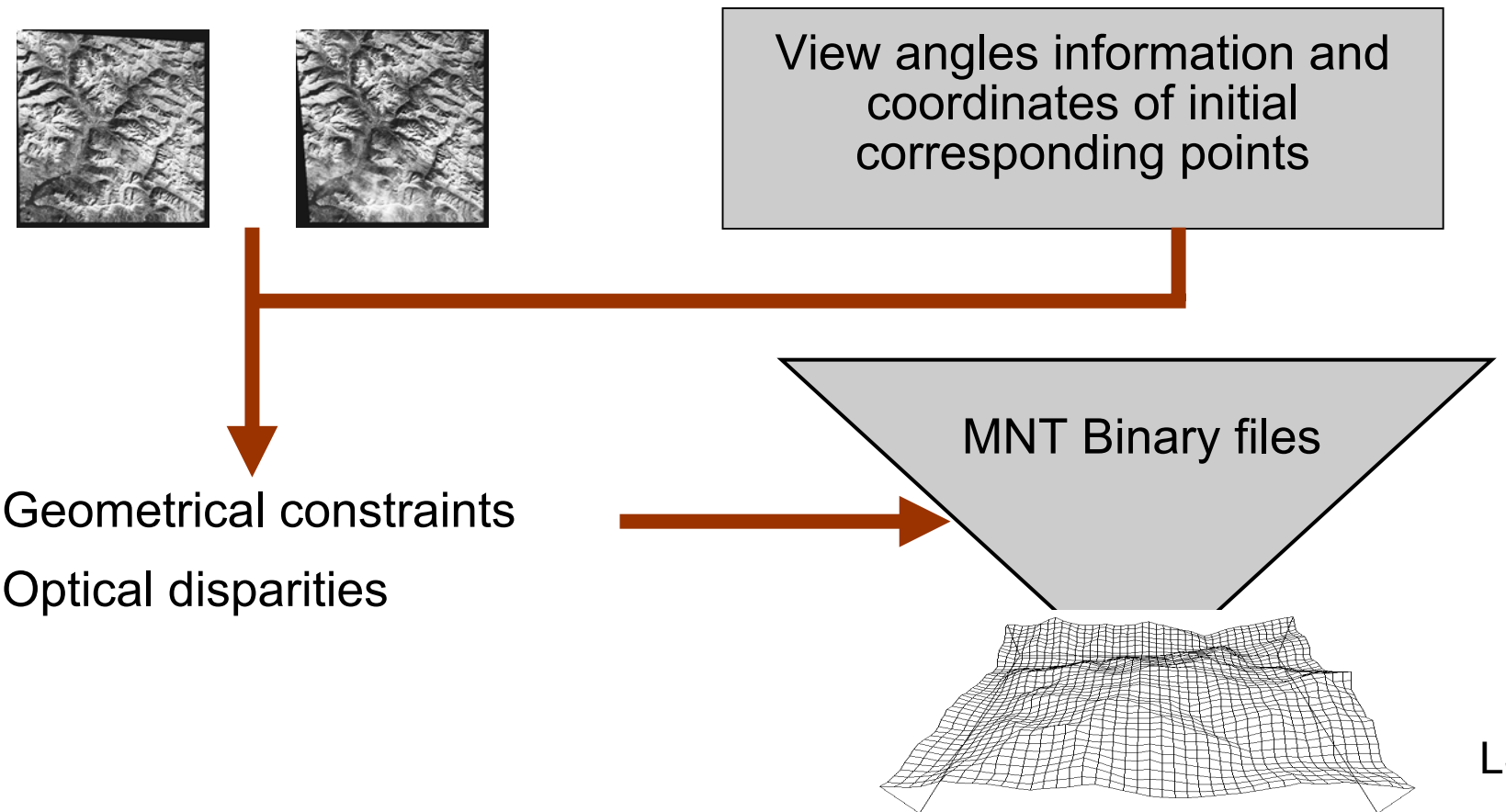

Vi

Some Target Applications

"My team has created a very innovative solution, but we're still looking for a problem to go with it."

Digital Elevation Models (MNT)

- Stereoscopic processing:
  - Maximal matching between the spots of both pictures.
  - Elevation computation.



Digital Elevation Models (MNT), cont.

Grid TLSE: expert site for sparse linear solvers

Tests for Large Systems of Equations

Coordinated by ENSEEIHT-IRIT, Toulouse

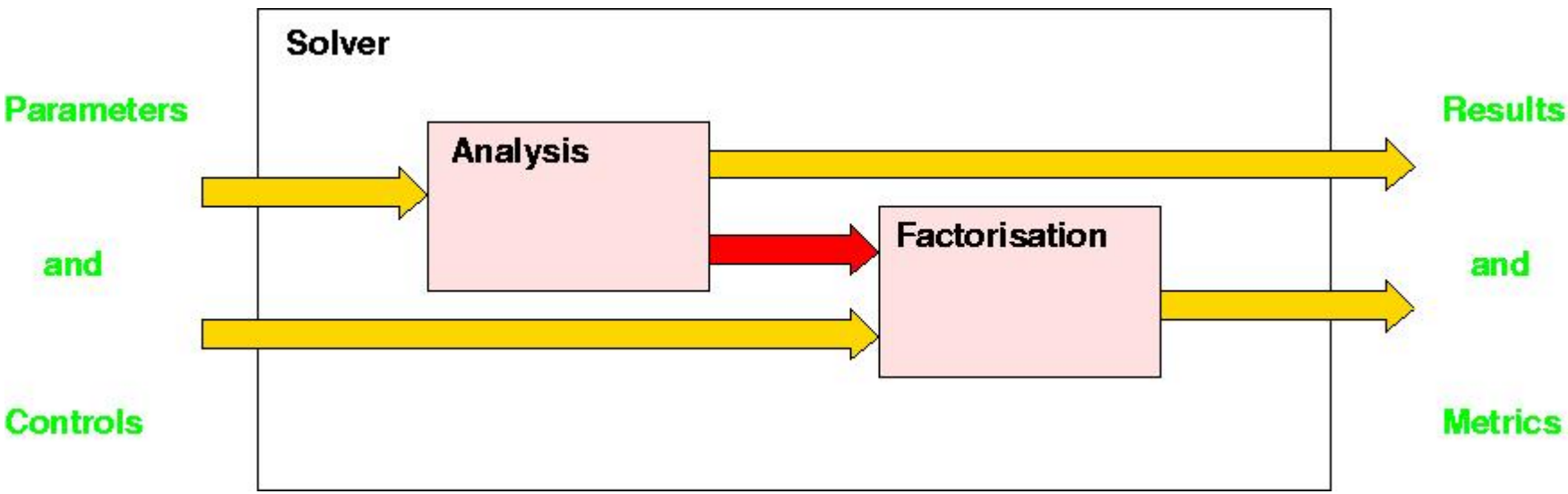
Funded by ACI GRID

- Goal
  - Provide a friendly test environment for expert and non-expert users of sparse direct linear algebra software
  - Easy access to software and tools, a wide range of computer architectures, matrix collections
  - On a user's specific problem, compare execution time / accuracy / memory usage / … of various sparse solvers
    - public domain … as well as commercial,
    - sequential … as well as parallel
    - Find best parameter values / reordering heuristics on a given problem

http://www.enseeiht.fr/lima/tlse

Request Examples

- Memory required to factor a matrix
- Error analysis as a function of the threshold pivoting value
- Minimum time on a given computer to factor a given unsymmetric matrix
- Which ordering heuristic is the best for solving a given problem



--- GRID-TLSE Project: Expertise results --- - Mozilla

File Edit View Go Bookmarks Tools Window Help

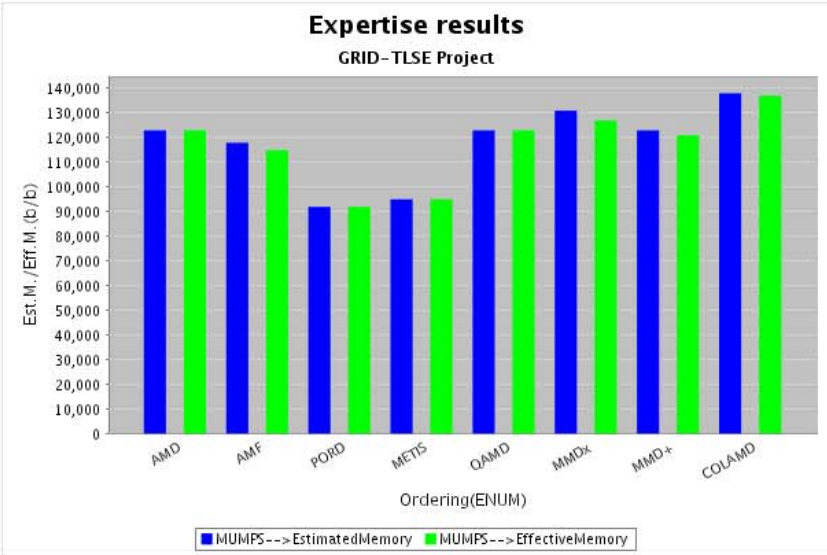
http://localhost:8080/GRID-TLSE/expertise_results.jsp?exp_id=34

Search

Home Bookmarks GRID-TLSE Google

Results for solver MUMPS

Matrix	Realisation	Ordering	EstimatedMemory	EffectiveMemory
/usr/local/share/matrices/DISK0/ex11.rua	MUMPS	AMD	123000	123000
/usr/local/share/matrices/DISK0/ex11.rua	MUMPS	AMF	118000	115000
/usr/local/share/matrices/DISK0/ex11.rua	MUMPS	PORD	92000	92000
/usr/local/share/matrices/DISK0/ex11.rua	MUMPS	METIS	95000	95000
/usr/local/share/matrices/DISK0/ex11.rua	MUMPS	QAMD	123000	123000
/usr/local/share/matrices/DISK0/ex11.rua	MUMPS	MMDx	131000	127000
/usr/local/share/matrices/DISK0/ex11.rua	MUMPS	MMD+	123000	121000
/usr/local/share/matrices/DISK0/ex11.rua	MUMPS	COLAMD	138000	137000



Expertise comments

There are no comments on this expertise.

- Make a comment -

--- GRID-TLSE Project: Ex
Inbox for hamerling@ensee
13:44
23/09/2004

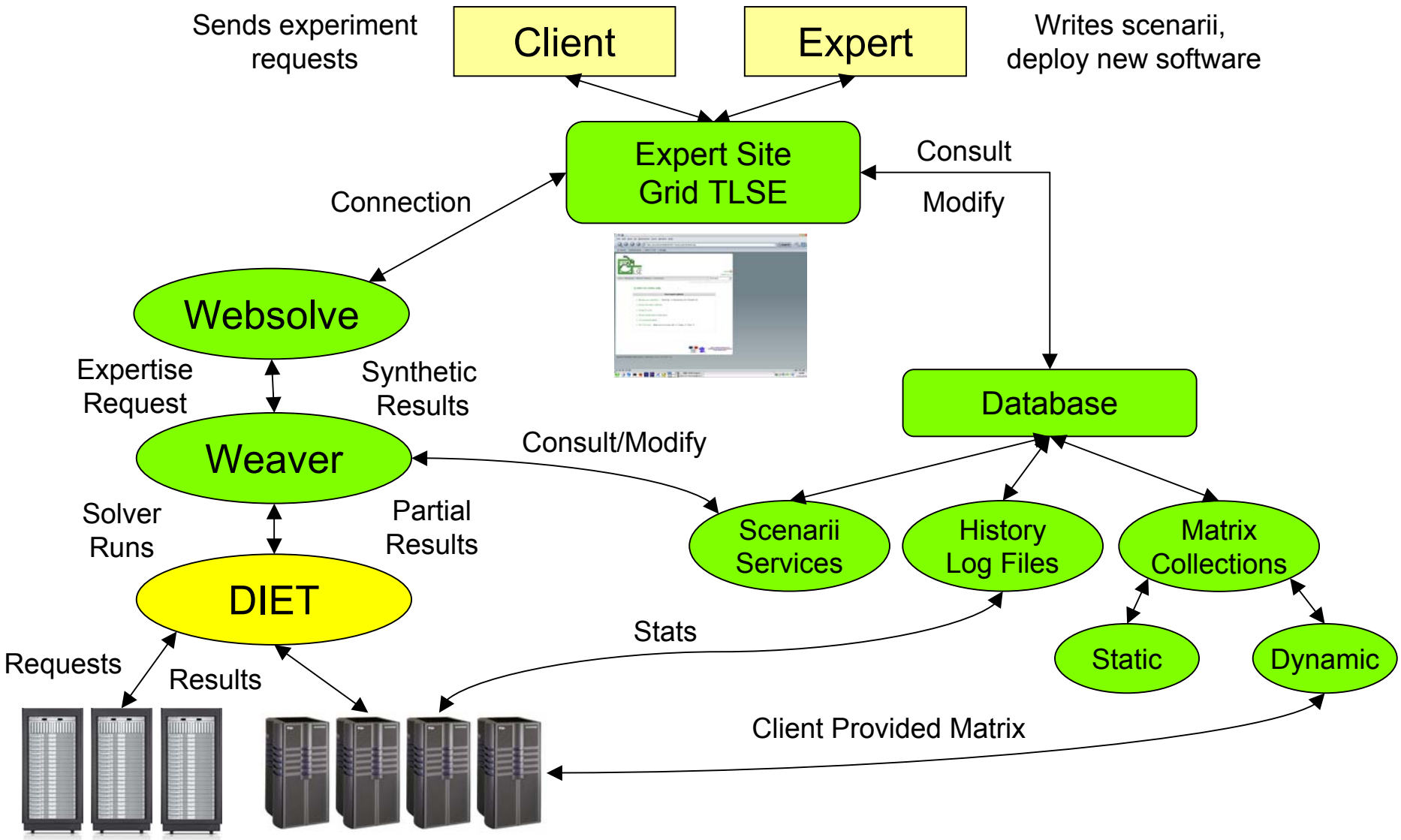
Why using a grid ?

- Sparse linear algebra software makes use of sophisticated algorithms for (pre/post)-processing the matrix
- Multiple parameters interfere for the efficient execution of sparse linear solvers
  - Ordering
  - Amount of memory
  - Architecture of the target computer
  - Available libraries
  - Determining the best combination of parameter values is multi-parametric problem
  - Combinatorial nature of these parameters
- The installation of any sparse solver library on a new architecture can be a nightmare !
- Testing different architectures
- Always using the latest version of each library

Is it realistic ?

- Time to send the data can be more important than the computation itself !
- But
  - Large number of independent requests
  - Time to answer is not critical
  - Data persistency between elementary requests easy to express
- Clear need for the users !
  - Managing software and hardware testing from a PSE

Architecture



Research Issues

- Sparse Linear Algebra
  - Automatically choosing the right parameters, the correct sequence of operations
  - Help the user as much as possible
- Scenarii Management
  - Generation and management of workflows
  - Need to be connected to the scheduling of requests/data management
- Interoperability
  - Connecting different libraries with different data formats
  - Meta-data
- Data Management
  - Leaving data in place as much as possible
  - Matrix collections
- DIET Research issues
  - Managing requests of different sizes with data dependences

Conclusion and Future Work

We did not talk about …

- Automatic deployment
  - Depending of the target architecture, the location of servers, clients, …
- Distributed scheduling
  - Plugin schedulers, relations with batch schedulers, application dependent scheduling
- Other applications
  - Simulation (physic, chemical eng., …), robotics, bioinformatic, geology, …
- Adding services
  - Registering new applications
- Performance evaluation
  - Routine/application cost, data (re)distribution, computation of the optimal number of processors used on the servers
- Fault tolerance
  - Agent, servers, checkpointing
- Platform monitoring
  - Distributed log management (LogService), post-mortem visualization (VizDIET)
- Security !
  - Authentication, communications, firewalls, …

Conclusions and future work

- GridRPC
  - Interesting approach for several applications
  - Flexible and efficient
  - Many interesting research issues (scheduling, data management, resource discovery and reservation, deployment, fault-tolerance, …)
- DIET
  - Scalable, open-source, and multi-application platform
  - Concentration on several issues like resource discovery, scheduling (distributed scheduling and plugin schedulers), deployment (GoDIET), performance evaluation (FAST and Freddy), monitoring (LogService and VizDIET), data management and replication (DTM and Juxmem)
  - Large scale validation on the Grid5000 platform
  - Demo @ SC2004
- TLSE
  - Help for the development of high performance sparse direct solvers
  - Opening the whole platform in 2005 (CSC 2005 ?)
  - Demo @ SC2004

http://www.grid5000.org/

http://graal.ens-lyon.fr/DIET (online demo)

Questions ?

http://graal.ens-lyon.fr/DIET