Grid-Aware Numerical Libraries

Using ScaLAPACK and PETSc on the Grid

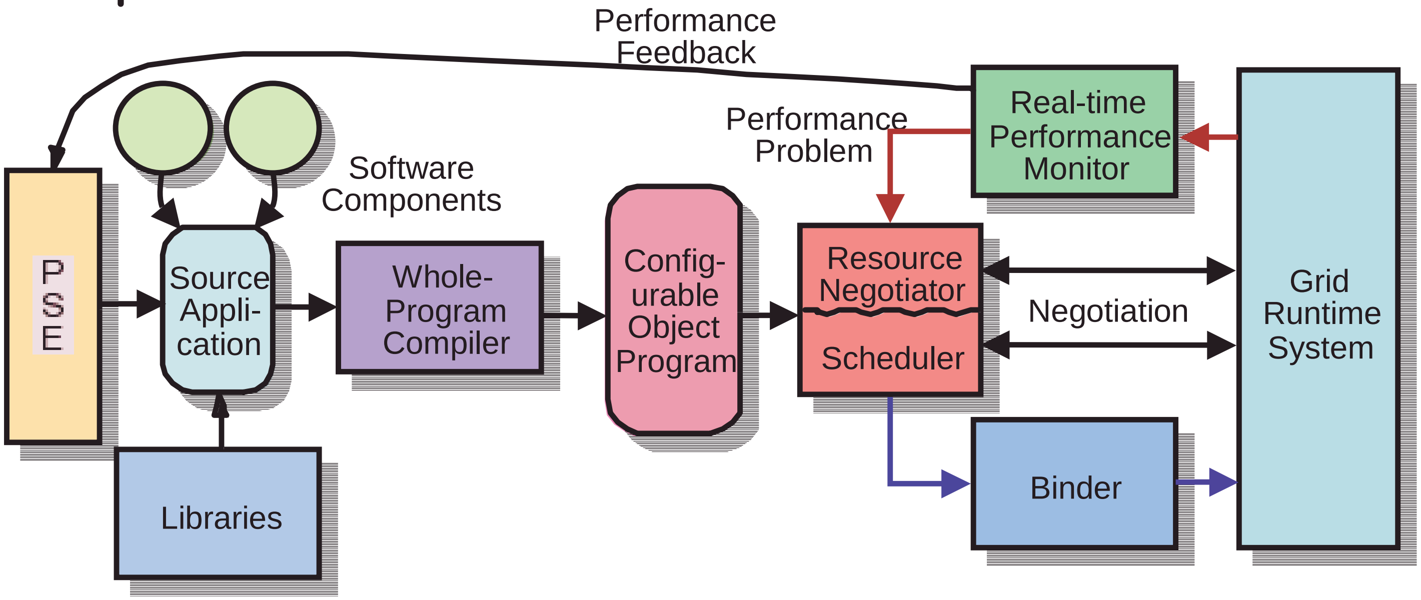
Jack Dongarra
University of Tennessee

http://hipersoft.rice.edu/stc_site_visit/talks/numlib.pdf

GrADS
Grid Application Development Software Project

Grid-Aware Numerical Libraries

- Using ScaLAPACK and PETSc on the Grid: Early Experiences



Performance Feedback

Software Components

Performance Problem

Real-time Performance Monitor

P S E

Source Appli-cation

Whole-Program Compiler

Config-urable Object Program

Resource Negotiator

Scheduler

Negotiation

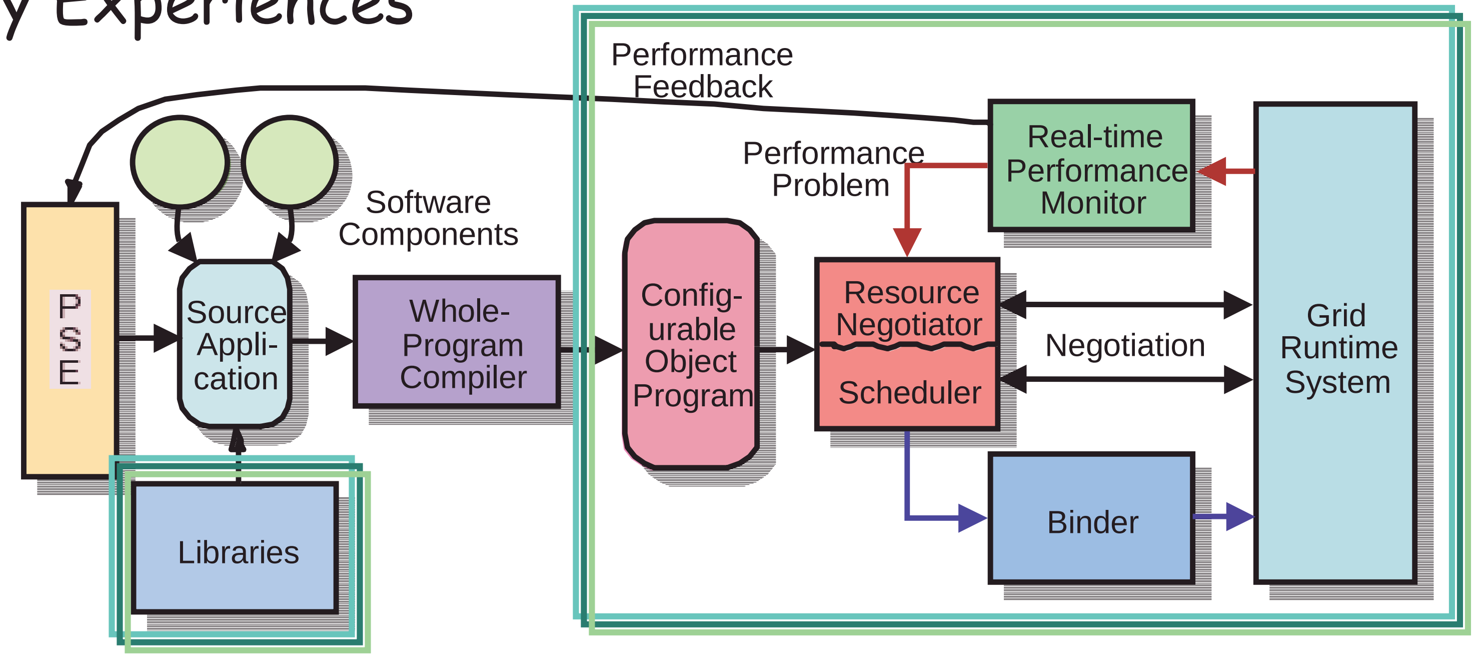
Grid Runtime System

Libraries

Binder

Grid-Aware Numerical Libraries

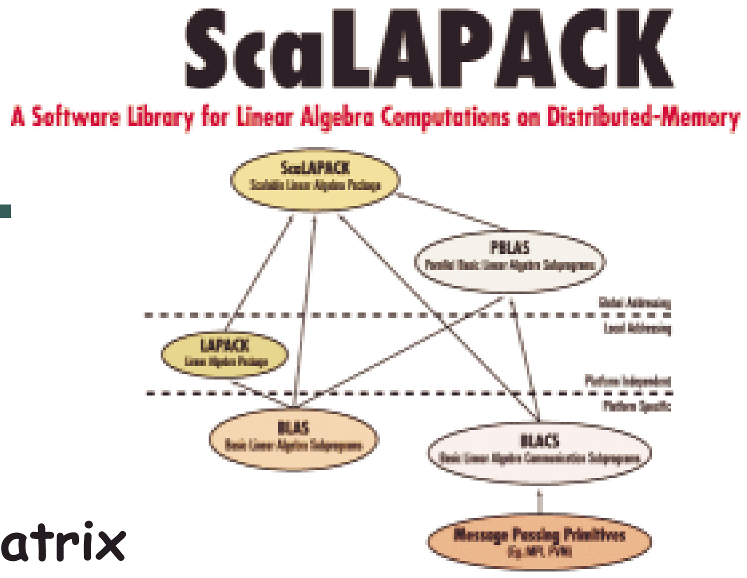
- Using ScaLAPACK and PETSc on the Grid: Early Experiences



In some sense ScaLAPACK not an ideal application for the Grid.

Expanded our understand how various GrADS component fit together.

ScaLAPACK


ScaLAPACK
A Software Library for Linear Algebra Computations on Distributed-Memory
ScaLAPACK
Scalable Linear Algebra Package
PBLAS
Parallel Basic Linear Algebra Subprograms
Global Addressing
Local Addressing
LAPACK
Linear Algebra Package
Platform Independent
Platform Specific
BLAS
Basic Linear Algebra Subprograms
BLACS
Basic Linear Algebra Communication Subprograms
Message Passing Primitives
(Eg. MPI, PVM)

- ScaLAPACK is a portable distributed memory numerical library
- Complete numerical library for dense matrix computations
- Designed for distributed parallel computing (MPP & Clusters) using MPI
- One of the first math software packages to do this
- Numerical software that will work on a heterogeneous platform
- Funding from DOE, NSF, and DARPA
- In use today by IBM, HP-Convex, Fujitsu, NEC, Sun, SGI, Cray, NAG, IMSL, …
  - Tailor performance & provide support

ScaLAPACK Grid Enabled

- Implement a version of a ScaLAPACK library routine that runs on the Grid.
  - Make use of resources at the user's disposal
  - Provide the best time to solution
  - Proceed without the user's involvement
- Make as few changes as possible to the numerical software.
- Assumption is that the user is already "Grid enabled" and runs a program that contacts the execution environment to determine where the execution should take place.

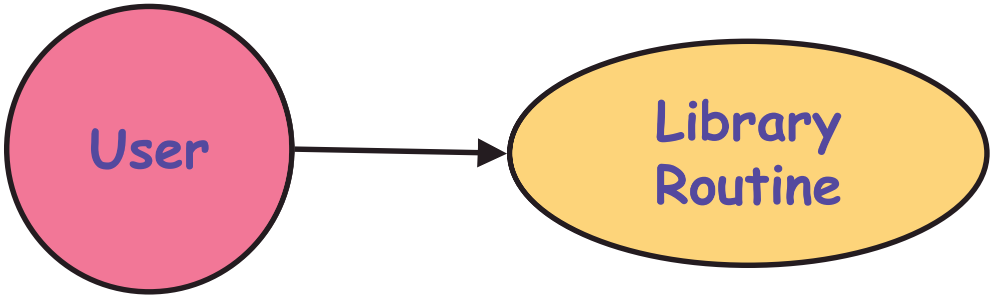
To Use ScaLAPACK a User Must:

- Download the package and auxiliary packages (like PBLAS, BLAS, BLACS, & MPI) to the machines.
- Write a SPMD program which
  - Sets up the logical 2-D process grid
  - Places the data on the logical process grid
  - Calls the numerical library routine in a SPMD fashion
  - Collects the solution after the library routine finishes
- The user must allocate the processors and decide the number of processes the application will run on
- The user must start the application
  - "mpirun –np *N* user_app"
    - Note: the number of processors is fixed by the user before the run, if problem size changes dynamically …
- Upon completion, return the processors to the pool of resources

GrADS Numerical Library

- Want to relieve the user of some of the tasks
- Make decisions on which machines to use based on the user's problem and the state of the system
  - Determinate machines that can be used
  - Optimize for the best time to solution
  - Distribute the data on the processors and collections of results
  - Start the SPMD library routine on all the platforms
  - Check to see if the computation is proceeding as planned
    - If not perhaps migrate application

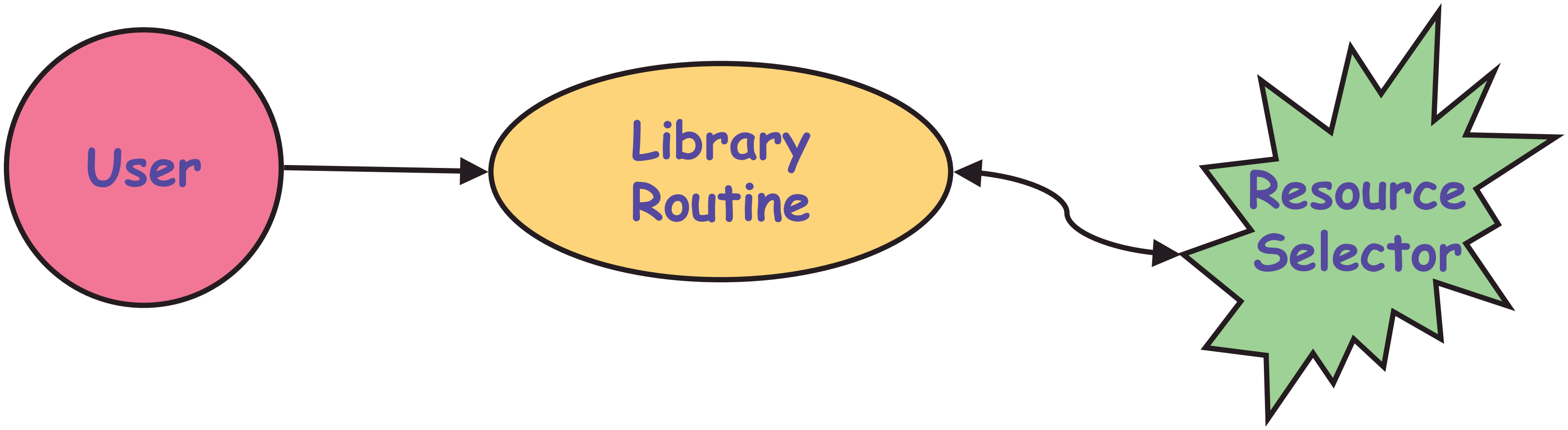
GrADS Library Sequence



- Has "crafted code" to make things work correctly and together.

Assumptions:
Autopilot Manager has been started
and
Globus is there.

Resource Selector



- Uses MDS and NWS to build an array of values for the machines that are available for the user.
  - 2 matrices (bw,lat) 2 arrays (cpu, memory available)
  - Matrix information is clique based
- On return from RS, Crafted Code filters information to use only machines that have the necessary software and are really eligible to be used.

Arrays of Values Generated by Resource Selector

- Clique based
  - 2 @ UT, UCSD, UIUC
    - Part of the MacroGrid
  - Full at the cluster level and the connections (clique leaders)
  - Bandwidth and Latency information looks like this.
  - Linear arrays for CPU and Memory
- Matrix of values are filled out to generate a complete, dense, matrix of values.
- At this point have a workable coarse grid.
  - Know what is available, the connections, and the power of the machines

x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x	x	x	x
x	x x x x x x x x x x x x x x x x	x	x
x	x	x x x x x x x x x x x x x x x x x x x x x x x x x	x
x	x	x	x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x

ScaLAPACK Performance Model

$$T(n,p) = C_f t_f + C_v t_v + C_m t_m$$

$C_f = \frac{2n^3}{3p}$ – Total number of floating-point operations per processor

$C_v = (3 + \frac{1}{4}\log_2 p)\frac{n^2}{\sqrt{p}}$ – Total number of data items communicated per processor

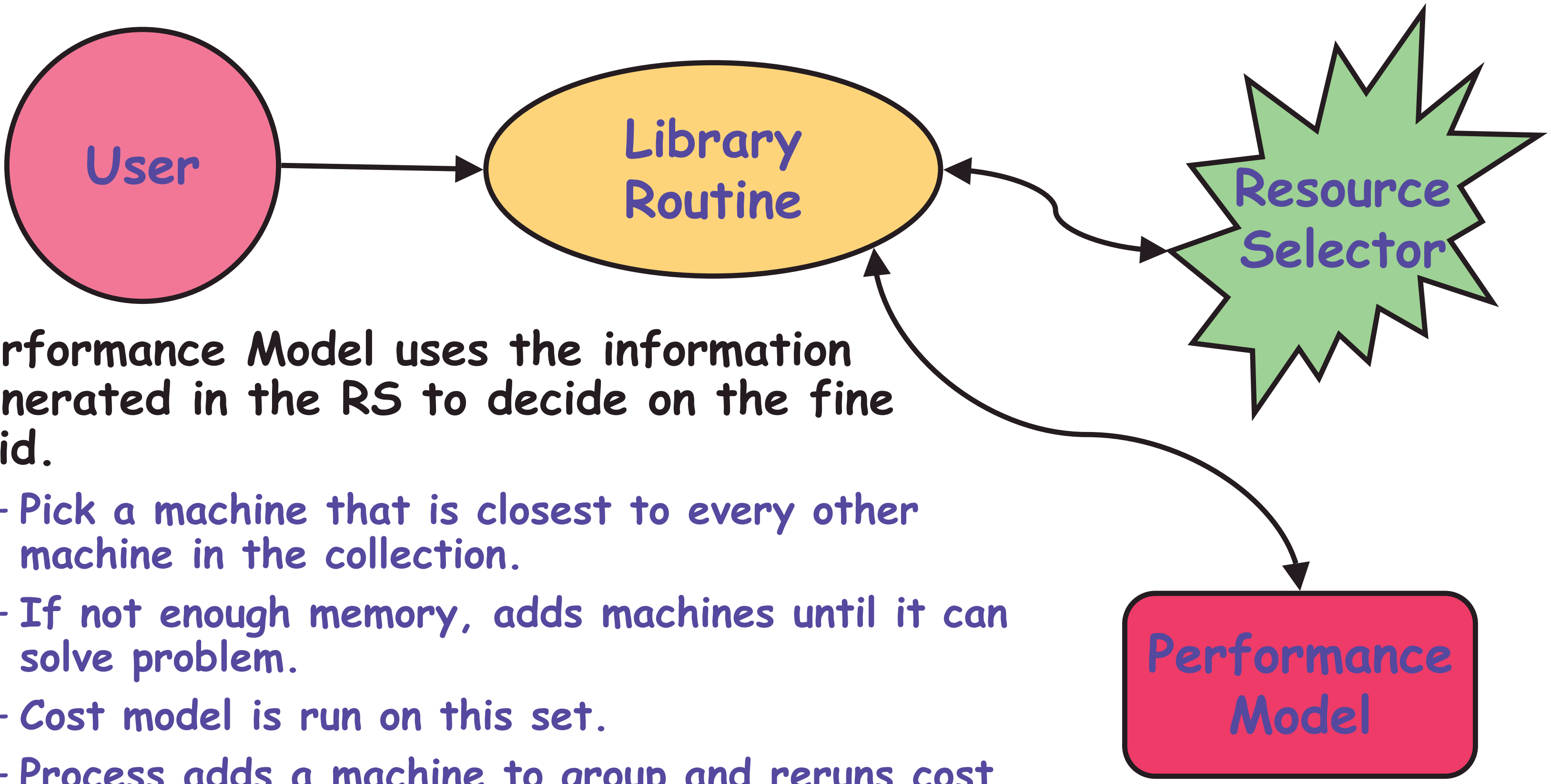
$C_m = n(6 + \log_2 p)$ – Total number of messages

t_f – Time per floating point operation

t_v – Time per data item communicated

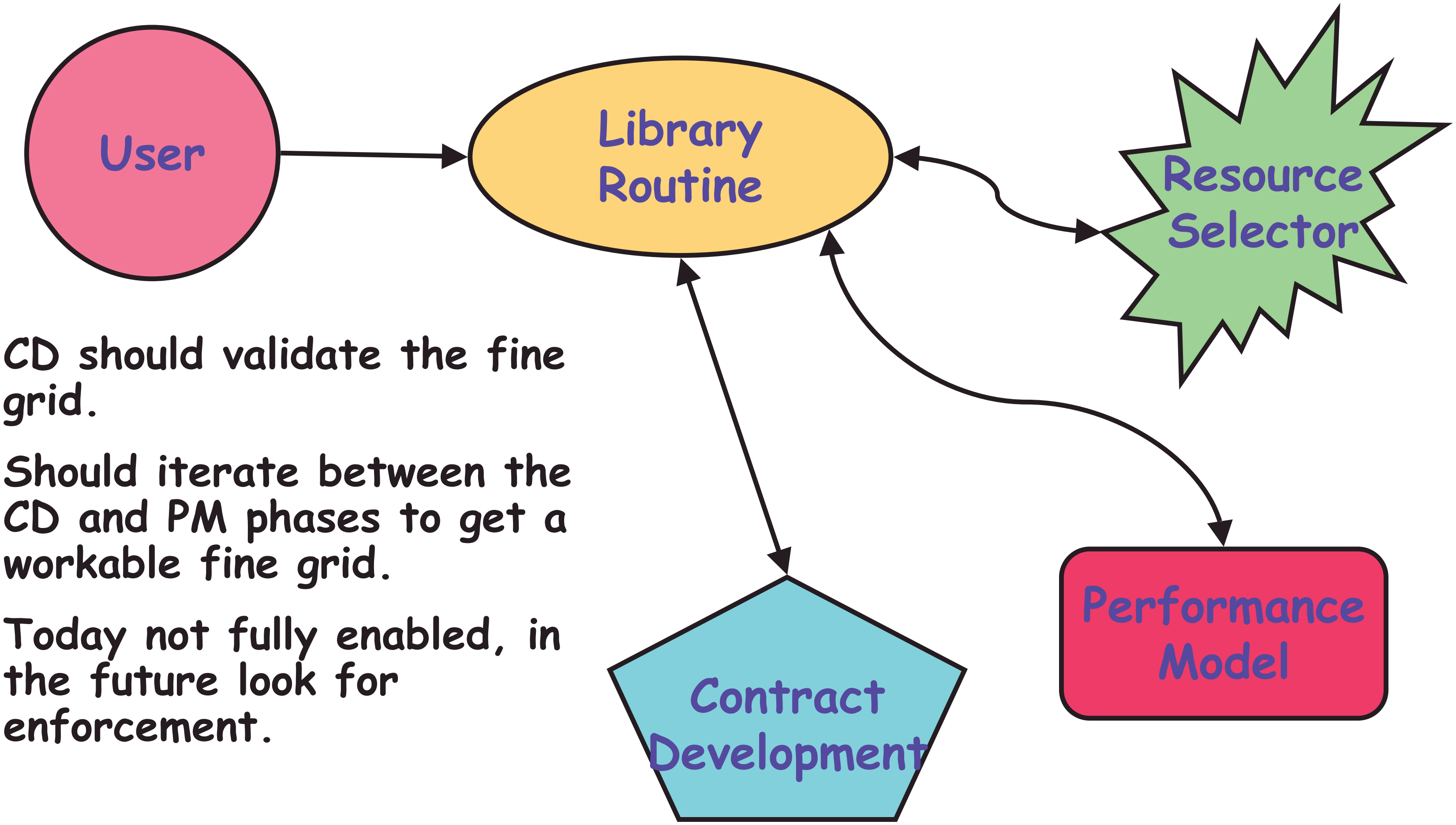
t_m – Time per message

Performance Model



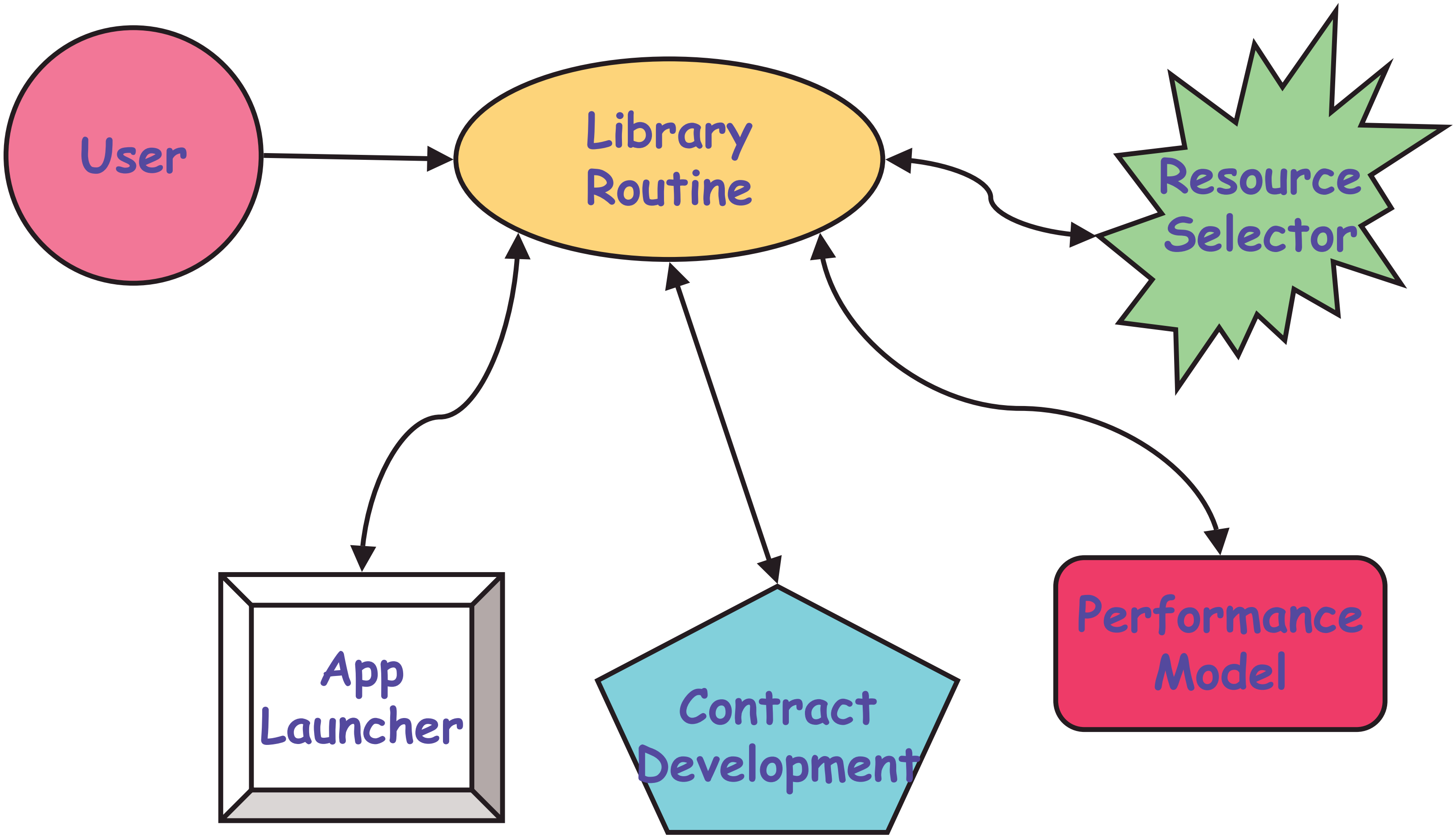
- **Performance Model uses the information generated in the RS to decide on the fine grid.**
  - Pick a machine that is closest to every other machine in the collection.
  - If not enough memory, adds machines until it can solve problem.
  - Cost model is run on this set.
  - Process adds a machine to group and reruns cost model.
  - If "better", iterate last step, if not stop.

Contract Development



- CD should validate the fine grid.
- Should iterate between the CD and PM phases to get a workable fine grid.
- Today not fully enabled, in the future look for enforcement.

Application Launcher



“mpirun –machinefile –globusrsl fine_grid grid_linear_solve”

Experimental Hardware / Software Grid

MacroGrid Testbed	TORC	CYPHER	OPUS
Type	Cluster 8 Dual Pentium III	Cluster 16 Dual Pentium III	Cluster 8 Pentium II
OS	Red Hat Linux 2.2.15 SMP	Debian Linux 2.2.17 SMP	Red Hat Linux 2.2.16
Memory	512 MB	512 MB	128 or 256 MB
CPU speed	550 MHz	500 MHz	265 – 448 MHz
Network	Fast Ethernet (100 Mbit/s) (3Com 3C905B) and switch (BayStack 350T) with 16 ports	Gigabit Ethernet (SK-9843) and switch (Foundry FastIron II) with 24 ports	Myrinet (LANai 4.3) with 16 ports each

- Globus version 1.1.3
- Autopilot version 2.3
- NWS version 2.0.pre2
- MPICH-G version 1.1.2
- ScaLAPACK version 1.6
- ATLAS/BLAS version 3.0.2
- BLACS version 1.1
- PAPI version 1.1.5
- GrADS' "Crafted code"

Independent components being put together and interacting

Performance Model Validation

	Opus14	Opus13	Opus16	Opus15	Torc4	Torc6	Torc7
mem(MB)	215	214	227	215	233	479	479
speed	270	270	270	270	330	330	330
load	1	0.99	1	0.99	1	1.04	0.87

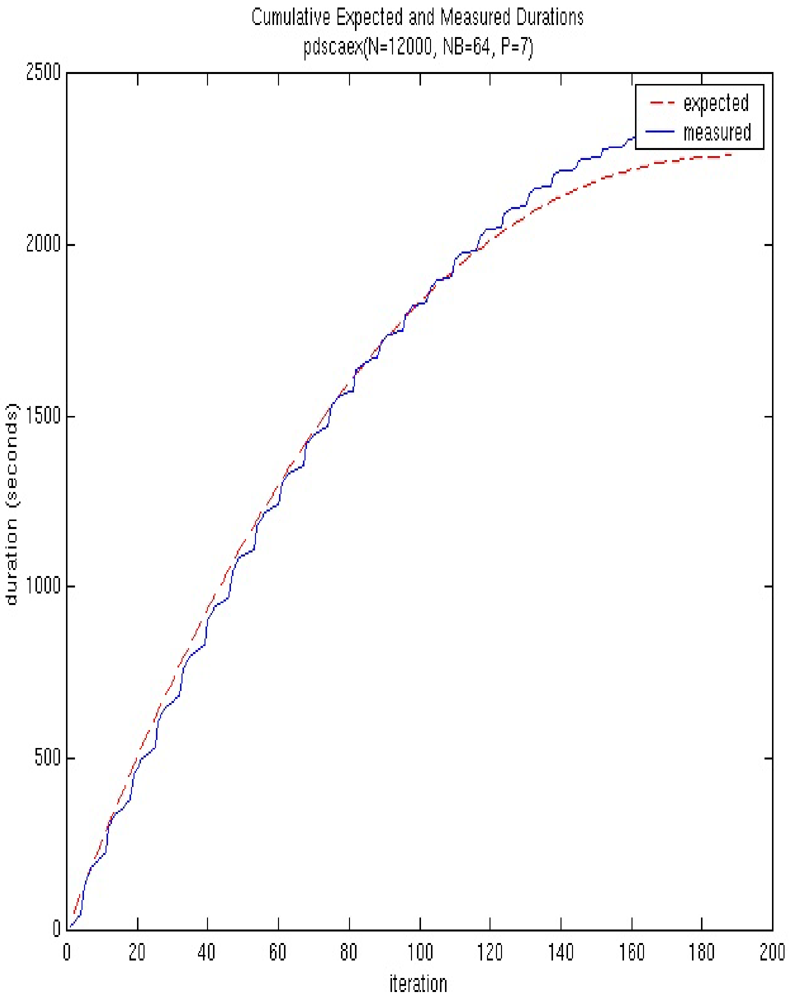
Speed = 60% of the peak

Latency	Opus14	Opus13	Opus16	Opus15	Torc4	Torc6	Torc7
Opus14	-1	0.24	0.29	0.26	83.78	83.78	83.78
Opus13	0.24	-1	0.24	0.23	83.78	83.78	83.78
Opus16	0.29	0.24	-1	0.23	83.78	83.78	83.78
Opus15	0.26	0.23	0.23	-1	83.78	83.78	83.78
Torc4	83.78	83.78	83.78	83.78	-1	0.31	0.31
Torc6	83.78	83.78	83.78	83.78	0.31	-1	0.31
Torc7	83.78	83.78	83.78	83.78	0.31	0.31	-1

Latency in msec

Bandwidth	Opus14	Opus13	Opus16	Opus15	Torc4	Torc6	Torc7
Opus14	-1	248.83	247.31	246.38	2.83	2.83	2.83
Opus13	248.83	-1	244.54	240.94	2.83	2.83	2.83
Opus16	247.31	244.54	-1	247.54	2.83	2.83	2.83
Opus15	246.38	240.94	247.54	-1	2.83	2.83	2.83
Torc4	2.83	2.83	2.83	2.83	-1	81.96	56.47
Torc6	2.83	2.83	2.83	2.83	81.96	-1	50.9
Torc7	2.83	2.83	2.83	2.83	56.47	50.9	-1

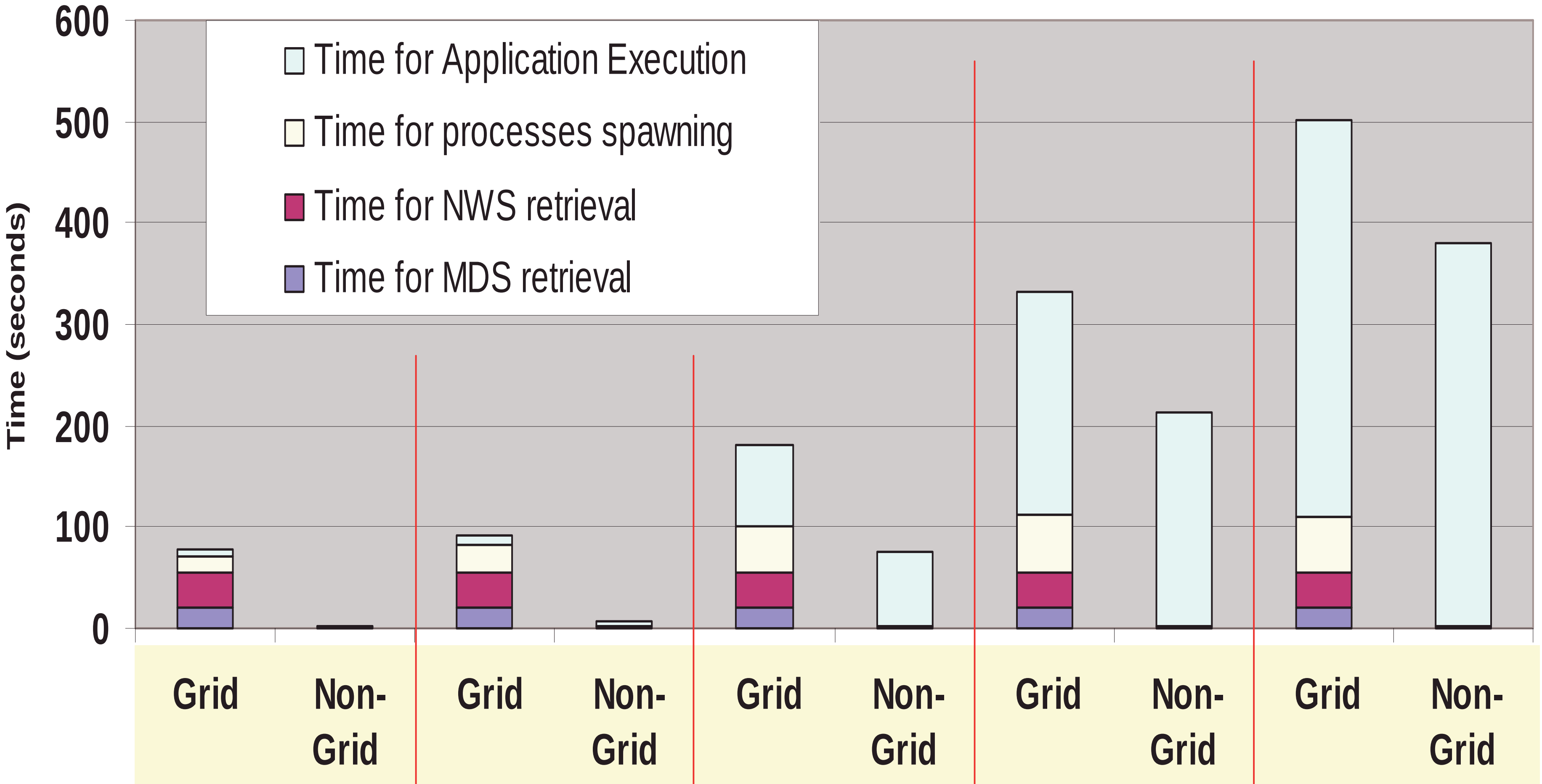
Bandwidth in Mb/s



This is for a refined grid

GrADS
Grid Application Development Software Project

Grid ScaLAPACK vs Non-Grid ScaLAPACK, Dedicated Torc machines



N=600, NB=40,
2 torc procs.
Ratio: 46.12

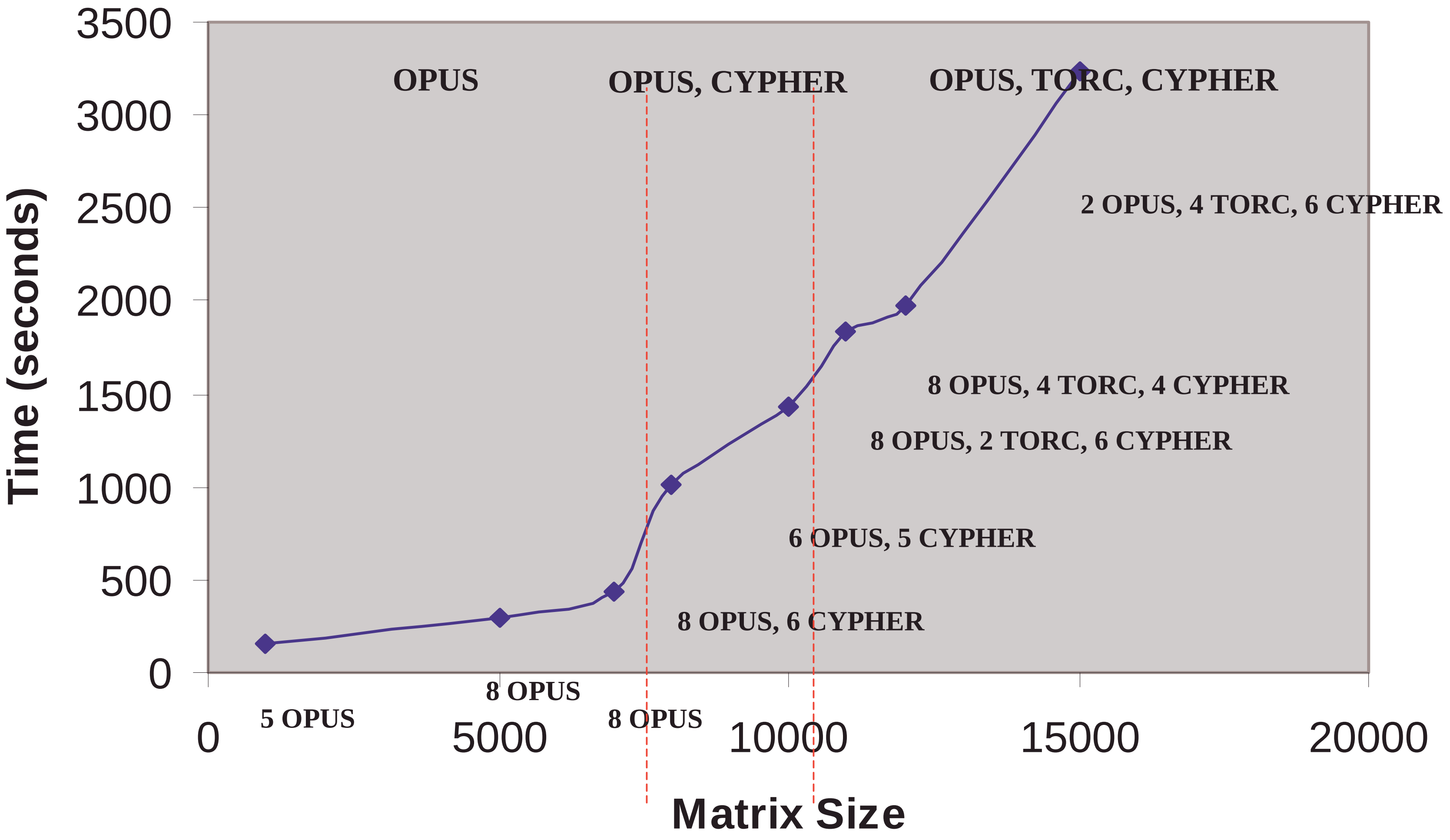
N=1500, NB=40,
4 torc procs.
Ratio: 15.03

N=5000, NB=40,
6 torc procs.
Ratio: 2.25

N=8000, NB=40,
8 torc procs.
Ratio: 1.52

N=10,000, NB=40,
8 torc procs.
Ratio: 1.29

ScaLAPACK across 3 Clusters



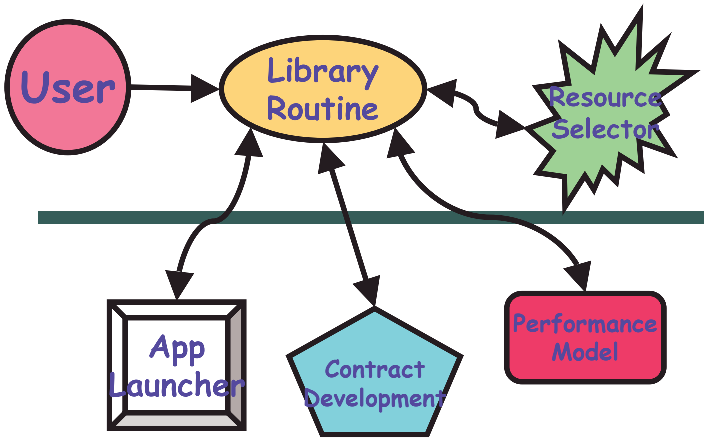
GrADS
Grid Application Development Software Project

Largest Problem Solved

- Matrix of size 30,000
  - 7.2 GB for the data
  - 32 processors to choose from UIUC and UT
    - Not all machines have 512 MBs, some little as 128 MBs
  - PM chose 17 machines in 2 clusters from UT
  - Computation took 84 minutes
    - 3.6 Gflop/s total
    - 210 Mflop/s per processor
    - ScaLAPACK on a cluster of 17 processors would get about 50% of peak
    - Processors are 500 MHz or 500 Mflop/s peak
    - For this grid computation 20% less than ScaLAPACK

PETSc and GrADS

- Portable, Extensible Toolkit for Scientific Computation
- Numerical software for scalable solutions for PDE’s
  - Toolkit for PDE’s
  - Portable and MPI based
- Large user base
  - One of the other numerical packages in wide use
- Has extensive sparse solvers
  - State of the art
- From Argonne National Lab
  - Balay, Gropp, Curfman, and Smith
- Applying the same ideas as was done in the ScaLAPACK example.



General Library Interface

- We have a start on an interface that will work, in general, with numerical libraries.
  - It's can be a "simple" operation to plug in other numerical routines/libraries.
  - Developing migration mechanisms for contract violations.
- Today a library writer needs to supply
  - Numerical Routine
  - Performance Model
- The rest of the framework can remain the same.

Conclusions

- Experiments are driving GrADS development
  - Hand developed leading to an automated design.
  - Exposes a number of areas for improvement
  - Very positive feed back to component developers with each experiment.
- CGrADS will automate much of the decisions in the Grid environment to provide best time to solution.
  - Adaptivity to the dynamic environment.
  - As the complexities of the Grid increase need to develop strategies for self adaptability.
- Developing a basic infrastructure for computational science applications and software in the Grid environment.
  - Lack of tools is hampering development today.
- As a team, we have tremendous experience and tools; and we are working together.
  - Coordinated integration
  - Critical mass of components
- Initiate community activities
  - Standards
- Opportunistic research
  - Seize on future opportunities to rapidly deploy and develop tools, standards, etc.

Strategy for Library Development on the Grid

- **Short term: 2 years**
  - Latency tolerance algorithms
  - Rules for self tuning
  - Historical information as input
  - Migration on contract violation
  - Explore the problems inherent in creating and deploying Grid applications, we will manually implement
  - Integration of various tools
  - Provide tools to community
  - Experimentally validate the library
- **Medium term: 3-5 years**
  - New library interfaces with higher-level abstractions together with performance contract and runtime support for adaptivity will be constructed.
  - Adaptive monitoring
  - Self tuning
  - Community activity for library and application mechanisms
  - Integration of various tools
  - Provide tools to community
  - Experimentally validate the library and framework
  - Work with commercial vendors to speed adoption of the ideas, technologies, and artifacts developed by CGrADS.
  - A complete set of Grid libraries, along with appropriate documentation and test cases.
- **Long term: 5-10 years**
  - High level algorithm description
  - Experimentally validate the library and framework
  - Continue work on new numerical algorithms capable of controlling large, complex heterogeneous and dynamic applications
  - Experimentally validate the complete GrADSoft framework, including integrated runtime systems, adaptive compilation systems, libraries, and PSE's