High Throughput Computing

Miron Livny
Computer Sciences Department
University of Wisconsin-Madison
miron@cs.wisc.edu

10 years ago we had

“The Grid”

The Grid: Blueprint for a New Computing Infrastructure

Edited by Ian Foster and Carl Kesselman
July 1998, 701 pages.

The grid promises to fundamentally change the way we think about and use computing. This infrastructure will connect multiple regional and national computational grids, creating a universal source of **pervasive and dependable** computing power that supports dramatically new classes of applications. The Grid provides a clear vision of what computational **grids** are, why we need them, who will use them, and how they will be programmed.

" … We claim that these **mechanisms**, although originally developed in the context of a cluster of workstations, are also applicable to computational **grids**. In addition to the required flexibility of services in these grids, a very important concern is that the system be **robust** enough to run in "**production mode**" continuously even in the face of component failures. … "

Miron Livny & Rajesh Raman, "*High Throughput Resource Management*", in "*The Grid: Blueprint for a New Computing Infrastructure*".

In the words of the CIO of Hartford Life

Resource: What do you expect to gain from grid computing? What are your main goals?

Severino: Well number one was scalability. …

Second, we obviously wanted **scalability with stability**. As we brought more servers and desktops onto the grid we didn't make it any less stable by having a bigger environment.

The third goal was cost savings. One of the most …

2,000 years ago we had the words of

Koheleth

son of David king in Jerusalem

The words of Koheleth son of David, king in Jerusalem ….

Only that shall happen
Which has happened,
Only that occur
Which has occurred;
There is nothing new
Beneath the sun!

Ecclesiastes Chapter 1 verse 9

35 years ago we had

The ALOHA network

One of the early computer networking designs, the ALOHA network was created at the University of Hawaii in **1970** under the leadership of Norman Abramson. Like the ARPANET group, the ALOHA network was built with DARPA funding. Similar to the ARPANET group, the ALOHA network was built to allow people in different locations to access the main computer systems. But while the ARPANET used leased phone lines, the ALOHA network used packet radio.

ALOHA was important because it used a shared medium for transmission. This revealed the need for more modern contention management schemes such as CSMA/CD, used by Ethernet. Unlike the ARPANET where each node could only talk to a node on the other end, in ALOHA everyone was using the same frequency. This meant that some sort of system was needed to control who could talk at what time. ALOHA's situation was similar to issues faced by modern Ethernet (non-switched) and Wi-Fi networks.

This shared transmission medium system generated interest by others. ALOHA's scheme was very simple. Because data was sent via a teletype the data rate usually did not go beyond **80 characters per second**. When two stations tried to talk at the same time, both transmissions were garbled. Then data had to be manually resent. ALOHA did not solve this problem, but it sparked interest in others, most significantly Bob Metcalfe and other researchers working at Xerox PARC. This team went on to create the Ethernet protocol.

30 years ago we had

Distributed Processing Systems

Claims for "benefits" provided by Distributed Processing Systems

P.H. Enslow, *"What is a Distributed Data Processing System?"* Computer, January 1978

- High Availability and Reliability
- High System Performance
- Ease of Modular and Incremental Growth
- Automatic Load and Resource Sharing
- Good Response to Temporary Overloads
- Easy Expansion in Capacity and/or Function

Definitional Criteria for a Distributed Processing System

P.H. Enslow and T. G. Saponas ""*Distributed and Decentralized Control in Fully Distributed Processing Systems*" Technical Report, 1981

- Multiplicity of resources
- Component interconnection
- Unity of control
- System transparency
- Component autonomy

Multiplicity of resources

The system should provide a number of assignable resources for any type of **service** demand. The greater the degree of replication of resources, the better the ability of the system to maintain high reliability and performance

Component interconnection

A Distributed System should include a communication subnet which interconnects the elements of the system. The transfer of information via the subnet should be controlled by a two-party, cooperative protocol (**loose coupling**).

Unity of Control

All the component of the system should be **unified** in their desire to achieve a **common goal**. This goal will determine the rules according to which each of these elements will be controlled.

System transparency

From the users point of view the set of resources that constitutes the Distributed Processing System acts like a "**single virtual machine**". When **requesting a service** the user should not require to be aware of the physical location or the instantaneous load of the various resources

Component autonomy

The components of the system, both the logical and physical, should be **autonomous** and are thus afforded the ability to refuse a request of service made by another element. However, in order to achieve the system's goals they have to interact in a **cooperative** manner and thus adhere to a common set of policies. These policies should be carried out by the control schemes of each element.

Challenges

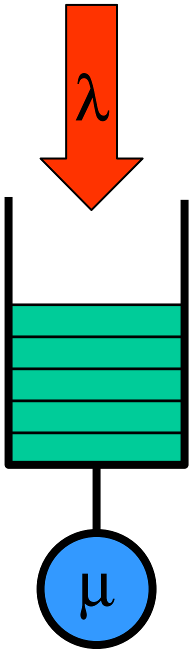
- Name spaces …
- Distributed ownership …
- Heterogeneity …
- Object addressing …
- Data caching …
- Object Identity …
- Trouble shooting …
- Circuit breakers …

24 years ago I wrote a Ph.D. thesis –

"Study of Load Balancing Algorithms for Decentralized Distributed Processing Systems"

http://www.cs.wisc.edu/condor/doc/livny-dissertation.pdf

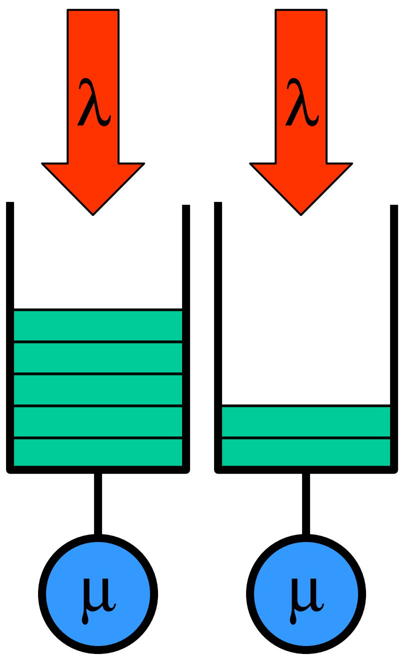
BASICS OF A M/M/1 SYSTEM



Expected # of customers is 1/(1-ρ), where (ρ = λ/μ) is the utilization

When utilization is 80%, you wait on the average 4 units for every unit of service

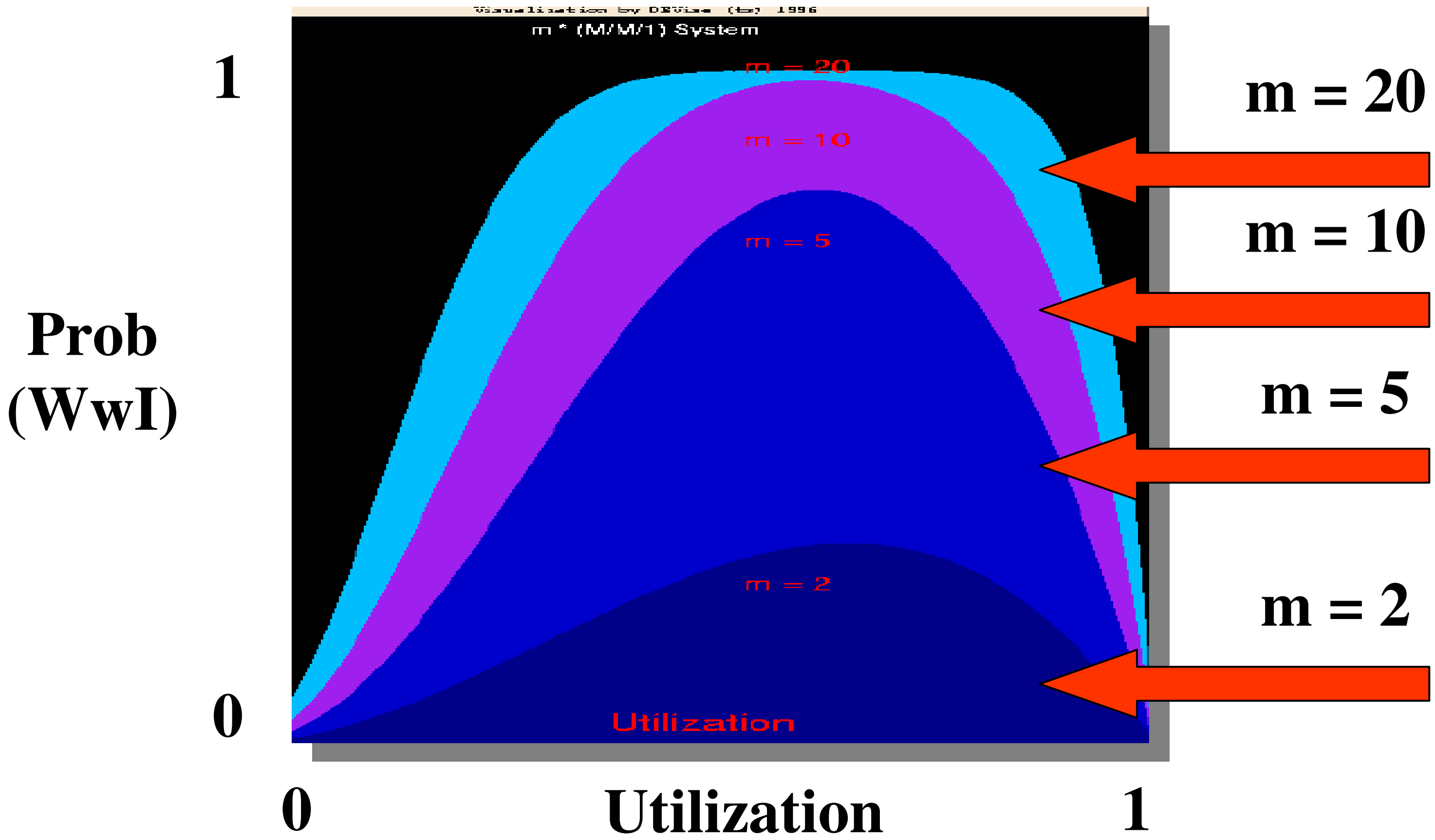
BASICS OF TWO M/M/1 SYSTEMS



When utilization is 80%, you wait on the average 4 units for every unit of service

When utilization is 80%, 25% of the time a customer is waiting for service while a server is idle

Wait while Idle (WwI) in m*M/M/1



" … Since the early days of mankind the primary motivation for the establishment of *communities* has been the idea that by being part of an organized group the capabilities of an individual are improved. The great progress in the area of inter-computer communication led to the development of means by which stand-alone processing sub-systems can be integrated into multi-computer '*communities*'. … "

Miron Livny, "*Study of Load Balancing Algorithms for Decentralized Distributed Processing Systems.*", Ph.D thesis, July 1983.

20 years ago we had

“Condor”

What Did We Learn From Serving a Quarter of a Million Batch Jobs on a Cluster of Privately Owned Workstations

1992

Miron Livny

Computer Sciences Department
University of Wisconsin — Madison
Madison, Wisconsin
{miron@cs.wisc.edu}

tive

capacity
accessible
iterface
rhead of
note

Global Scientific Computing via a Flock of Condors

CERN 92

Miron Livny

Computer Sciences Department
University of Wisconsin — Madison
Madison, Wisconsin
{miron@cs.wisc.edu}

MISSION

Give scientists effective and efficient access to large amounts of cheap (if possible free) CPU cycles and main memory storage

THE CHALLENGE

How to turn existing privetly owned clusters of *workstations*, *farms*, *multiprocessors*, and *supercomputers* into an efficient and effective Global Computing Environment?

In other words, how to minimize wait while idle?

APPROACH

Use wide-area networks to transfer batch jobs between Condor systems

- Boundaries of each Condor system will be determined by physical or administrative considerations

TWO EFFORTS

☐ ***UW CAMPUS***

Condor systems at Engineering, Statistics, and Computer Sciences

☐ ***INTERNATIONAL***

We have started a collaboration between CERN-SMC-NIKHEF-Univ. of Amsterdam, and University of Wisconsin-Madison

We are still very busy

1986-2006

Celebrating

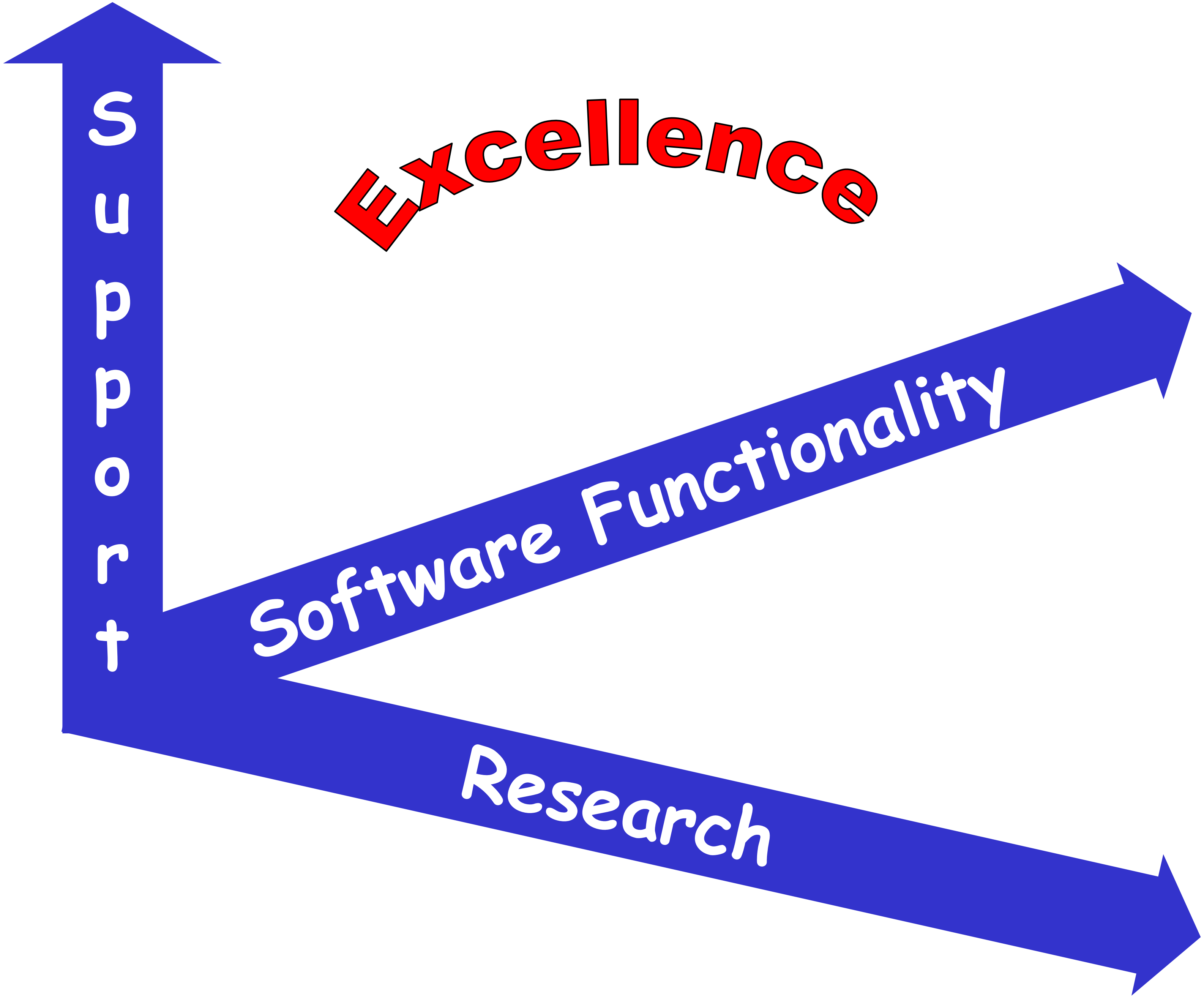
20 years since we first installed Condor in our department

Welcome to CW 2007!!!

The Condor Project (Established '85)

Distributed Computing **research** performed by a team of **~40** faculty, full time staff and students who

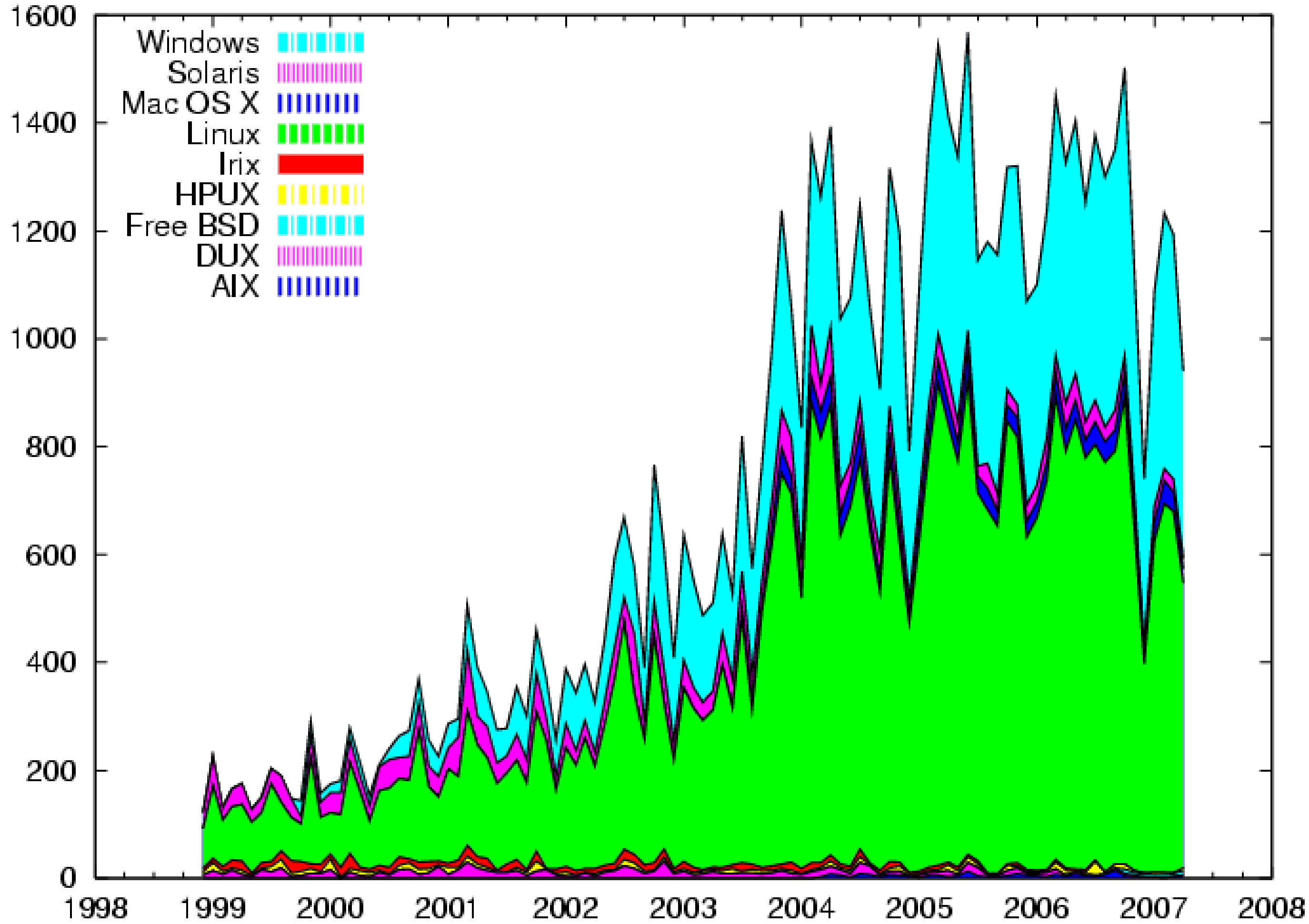
- face **software/middleware engineering** challenges in a UNIX/Linux/Windows/OS X environment,
- involved in national and international **collaborations**,
- interact with **users** in academia and industry,
- maintain and support a distributed **production** environment (more than 4000 CPUs at UW),
- and educate and train **students**.


Excellence
Support
Software Functionality
Research

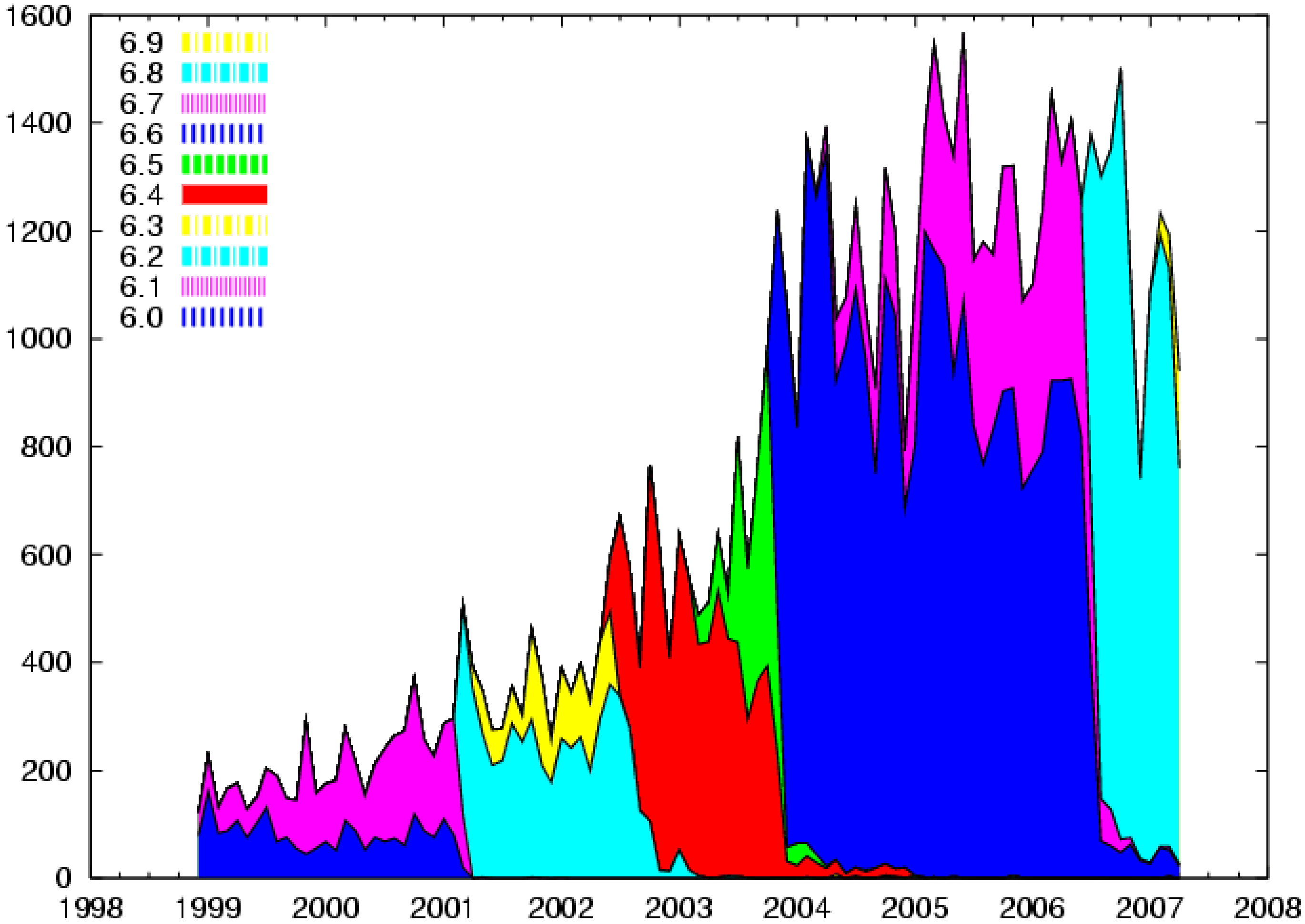
Main Threads of Activities

- Distributed Computing Research – develop and evaluate new concepts, frameworks and technologies
- Keep Condor "flight worthy" and support our users
- The Open Science Grid (OSG) – build and operate a national High Throughput Computing infrastructure
- The Grid Laboratory Of Wisconsin (GLOW) – build, maintain and operate a distributed computing and storage infrastructure on the UW campus The NSF Middleware Initiative
- Develop, build and operate a national Build and Test facility powered by Metronome
-

Downloads per month



Downloads per month



Grid Laboratory of Wisconsin

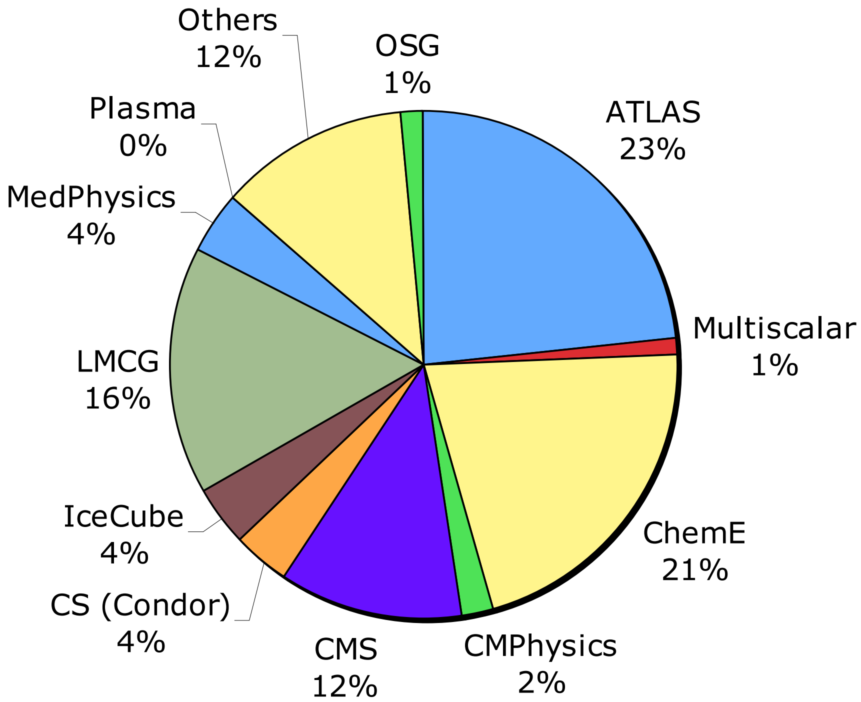
2003 Initiative funded by NSF(MIR)/UW at ~ $1.5M

Six Initial GLOW Sites

- Computational Genomics, Chemistry
- Amanda, Ice-cube, Physics/Space Science
- High Energy Physics/CMS, Physics
- Materials by Design, Chemical Engineering
- Radiation Therapy, Medical Physics
- Computer Science

Diverse users with different deadlines and usage patterns.

GLOW Usage 4/04-12/06



Over 23.4M

CPU hours served!

The search for SUSY

- Sanjay Padhi is a UW Chancellor Fellow who is working at the group of Prof. Sau Lan Wu at CERN
- Using Condor Technologies he established a "grid access point" in his office at CERN
- Through this access-point he managed to harness in 3 month (12/05-2/06) more that 500 CPU years from the LHC Computing Grid (LCG) the Open Science Grid (OSG) and UW Condor resources

Some Reports From the Field

- Condor at Micron
- Condor at BNL
- Condor at JPMorgan
- Condor at the Hartford
- Most production grid jobs (EGEE and OSG) are managed by Condor-G and related technologies

Integrating Linux Technology with Condor

Kim van der Riet

Principal Software Engineer

What will Red Hat be doing?

Red Hat will be investing into the Condor project locally in Madison WI, in addition to driving work required in upstream and related projects. This work will include:

- Engineering on Condor features & infrastructure
  - Should result in tighter integration with related technologies
  - Tighter kernel integration
- Information transfer between the Condor team and Red Hat engineers working on things like Messaging, Virtualization, etc.
- Creating and packaging Condor components for Linux distributions
- Support for Condor packaged in RH distributions

All work goes back to upstream communities, so this partnership will benefit all.

- Shameless plug: *If you want to be involved, Red Hat is hiring...*

High Throughput Computing

We first introduced the distinction between High Performance Computing (HPC) and High Throughput Computing (HTC) in a seminar at the NASA Goddard Flight Center in July of **1996** and a month later at the European Laboratory for Particle Physics (CERN). In June of 1997 HPCWire published an interview on High Throughput Computing.

Why HTC?

For many experimental scientists, scientific progress and quality of research are strongly linked to computing throughput. In other words, they are less concerned about instantaneous computing power. Instead, what matters to them is the amount of computing they can harness over a month or a year --- they measure computing power in units of scenarios per day, wind patterns per week, instructions sets per month, or crystal configurations per year.

High Throughput Computing is a 24-7-365 activity

$$\text{FLOPY} \neq (60*60*24*7*52)*\mathit{FLOPS}$$

Obstacles to HTC

› Ownership Distribution	(Sociology)
› Customer Awareness	(Education)
› Size and Uncertainties	(Robustness)
› Technology Evolution	(Portability)
› Physical Distribution	(Technology)

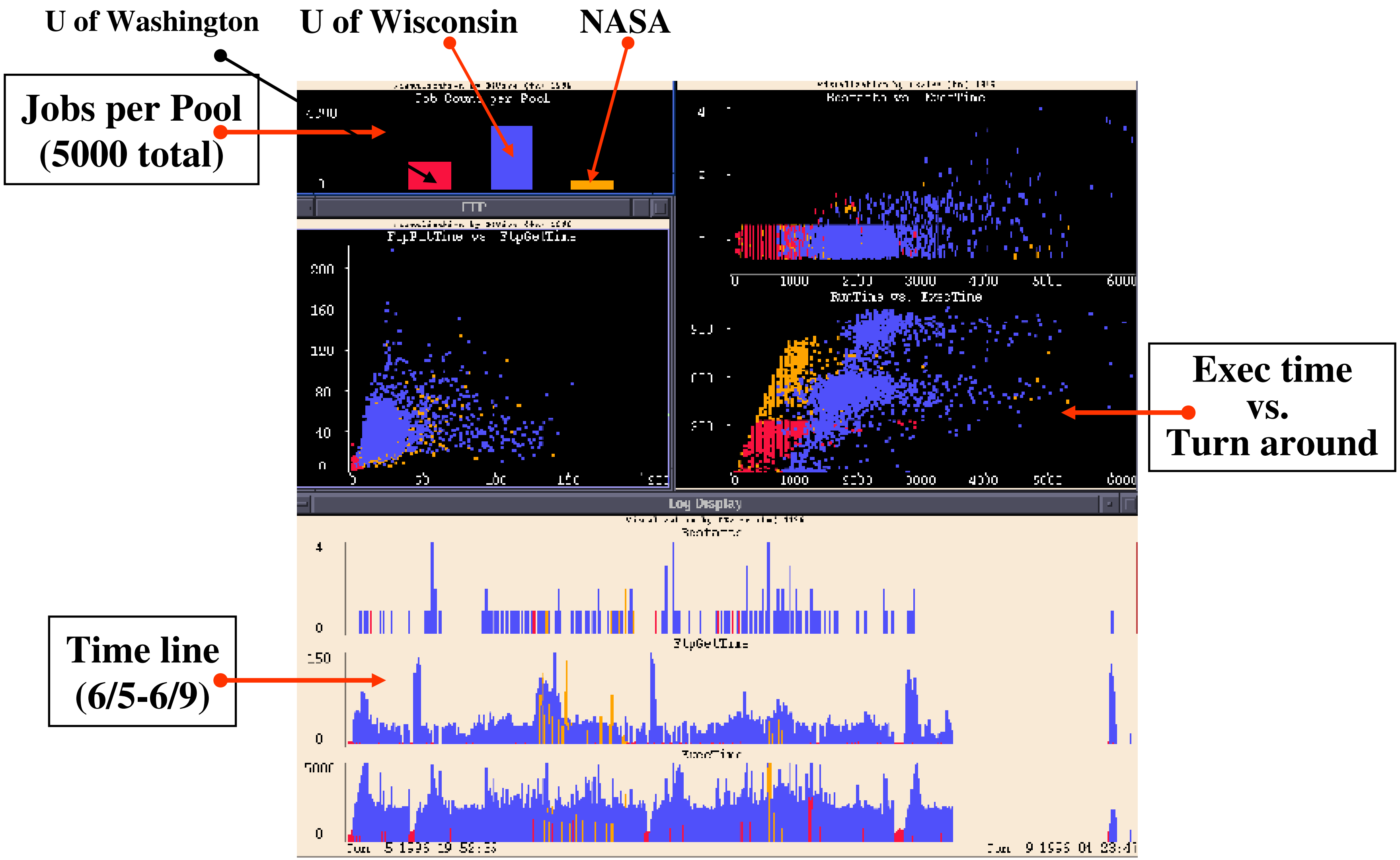
Focus on the problems that are **unique** to HTC not the latest/greatest **technology**

HTC on the Internet (1993)

Retrieval of atmospheric temperature and humidity profiles from 18 years of data from the TOVS sensor system.

- 200,000 images
- ~5 minutes per image

Executed on Condor pools at the University of Washington, University of Wisconsin and NASA. Controlled by DBC (Distributed Batch Controller). Execution log visualized by DEVise



U of Washington
U of Wisconsin
NASA
Jobs per Pool
(5000 total)
Exec time
vs.
Turn around
Time line
(6/5-6/9)
Job Count per Pool
Log Display

IBM Systems and Technology Group

High Throughput Computing on Blue Gene

IBM Rochester: Amanda Peters, Tom Budnik

With contributions from:
IBM Rochester: Mike Mundy, Greg Stewart, Pat McCarthy
IBM Watson Research: Alan King, Jim Sexton
UW-Madison Condor: Greg Thain, Miron Livny, Todd Tannenbaum

Condor and IBM Blue Gene Collaboration

- **Both IBM and Condor teams engaged in adapting code to bring Condor and Blue Gene technologies together**

- **Initial Collaboration (Blue Gene/L)**
  - Prototype/research Condor running HTC workloads on Blue Gene/L
    - Condor developed dispatcher/launcher running HTC jobs
    - Prototype work for Condor being performed on Rochester On-Demand Center Blue Gene system

- **Mid-term Collaboration (Blue Gene/L)**
  - Condor supports HPC workloads along with HTC workloads on Blue Gene/L

- **Long-term Collaboration (Next Generation Blue Gene)**
  - I/O Node exploitation with Condor
  - Partner in design of HTC services for Next Generation Blue Gene
    - Standardized launcher, boot/allocation services, job submission/tracking via database, etc.
  - Study ways to automatically switch between HTC/HPC workloads on a partition
  - Data persistence (persisting data in memory across executables)
    - Data affinity scheduling
  - Petascale environment issues

10 years ago we had

"The Grid"

Introduction

"The term "the Grid" was coined in the mid 1990s to denote a proposed **distributed computing** infrastructure for advanced science and engineering [27]. Considerable progress has since been made on the construction of such an infrastructure (e.g., [10, 14, 36, 47]) but the term "Grid" has also been conflated, at least in popular perception, to embrace everything from advanced networking to artificial intelligence. One might wonder if the term has any real substance and meaning. Is there really a distinct "Grid problem" and hence a need for new "Grid technologies"? If so, what is the nature of these technologies and what is their domain of applicability? While numerous groups have interest in Grid concepts and share, to a significant extent, a common vision of Grid architecture, we do not see consensus on the answers to these questions."

"The Anatomy of the Grid - Enabling Scalable Virtual Organizations" Ian Foster, Carl Kesselman and Steven Tuecke 2001.

Global Grid Forum (March 2001)

The Global Grid Forum (***Global GF***) is a community-initiated forum of individual researchers and practitioners working on **distributed computing**, or "grid" technologies. Global GF focuses on the promotion and development of Grid technologies and applications via the development and documentation of "best practices," implementation guidelines, and standards with an emphasis on rough consensus and running code.

Global GF efforts are also aimed at the development of a broadly based Integrated Grid Architecture that can serve to guide the research, development, and deployment activities of the emerging Grid communities. Defining such an architecture will advance the Grid agenda through the broad deployment and adoption of fundamental basic services and by sharing code among different applications with common requirements.

Wide-area distributed computing, or "grid" technologies, provide the foundation to a number of large scale efforts utilizing the global Internet to build distributed computing and communications infrastructures..

Summary

"We have provided in this article a concise statement of the "Grid problem," which we define as **controlled resource sharing and coordinated resource use in dynamic, scalable virtual organizations**. We have also presented both requirements and a framework for a Grid architecture, identifying the principal functions required to enable sharing within **VOs** and defining key relationships among these different functions."

"The Anatomy of the Grid - Enabling Scalable Virtual Organizations" Ian Foster, Carl Kesselman and Steven Tuecke 2001.

What makes an "O" a "VO"?

What is new beneath the sun?

- **Distributed ownership** – who defines the "system's common goal"? No more one system.
- **Many administrative domains** – authentication, authorization and trust.
- **Demand is real** – many have computing needs that can not be addressed by centralized locally owned systems
- **Expectations are high** – Regardless of the question, distributed technology is "the" answer.
- **Distributed computing is once again "in".**

Benefits to Science

- Democratization of Computing – "you do not have to be a SUPER person to do SUPER computing." (accessibility)
- Speculative Science – "Since the resources are there, lets run it and see what we get." (unbounded computing power)
- Function shipping – "Find the image that has a red car in this 3 TB collection." (computational mobility)

The NUG30 Quadratic Assignment Problem (QAP)

$$\min_{p \in \Pi} \sum_{i=1}^{30} \sum_{j=1}^{30} a_{ij} b_{p(i)p(j)}$$

Solved!

(4 Scientists +

1 Linux Box)

NUG30 Personal Grid …

Managed by **one** Linux box at Wisconsin

Flocking:
- -- the main Condor pool at Wisconsin (500 processors)
- -- the Condor pool at Georgia Tech (284 Linux boxes)
- -- the Condor pool at UNM (40 processors)
- -- the Condor pool at Columbia (16 processors)
- -- the Condor pool at Northwestern (12 processors)
- -- the Condor pool at NCSA (65 processors)
- -- the Condor pool at INFN Italy (54 processors)

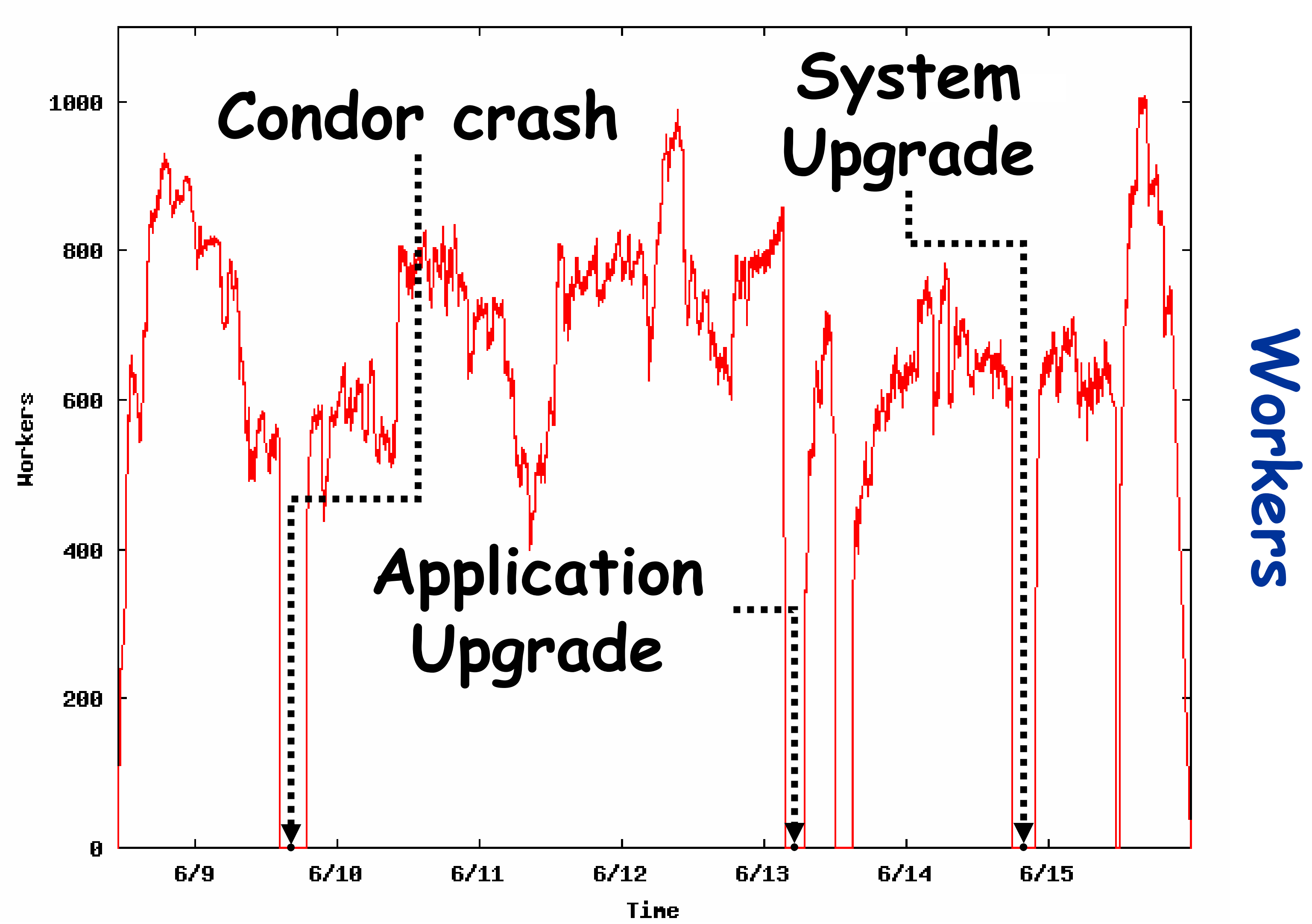
Glide-in:
- -- Origin 2000 (through LSF) at NCSA. (512 processors)
- -- Origin 2000 (through LSF) at Argonne (96 processors)

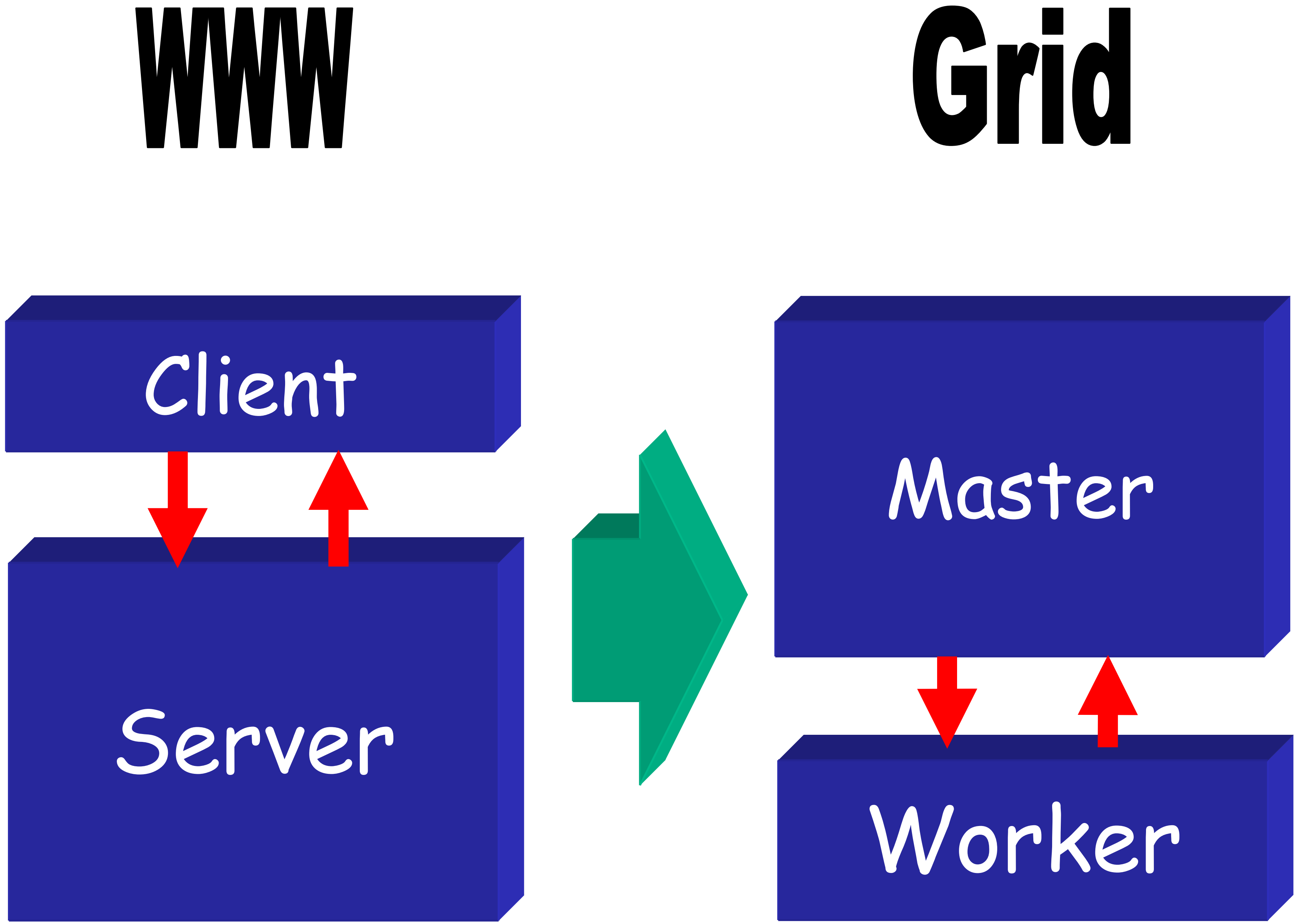
Hobble-in:
- -- Chiba City Linux cluster (through PBS) at Argonne (414 processors).

Solution Characteristics.

Scientists	**4**
Workstations	**1**
Wall Clock Time	6:22:04:31
Avg. # CPUs	653
Max. # CPUs	1007
Total CPU Time	Approx. 11 years
Nodes	11,892,208,412
LAPs	574,254,156,532
Parallel Efficiency	92%

The NUG30 Workforce




WWW
Grid
Client
Server
Master
Worker

" … Grid computing is a partnership between clients and servers. Grid clients have more responsibilities than traditional clients, and must be equipped with powerful mechanisms for dealing with and recovering from failures, whether they occur in the context of remote execution, work management, or data output. When clients are powerful, servers must accommodate them by using careful protocols.… "

Douglas Thain & Miron Livny, *"Building Reliable Clients and Servers", in "The Grid: Blueprint for a New Computing Infrastructure", 2nd edition*

Being a Master

Customer "delegates" task(s) to the master who is responsible for:

- Obtaining **allocation** of resources
- Deploying and managing workers on allocated resources
- **Delegating** work unites to deployed workers
- Receiving and processing results
- Delivering results to customer

Master must be …

- Persistent – work and results must be safely recorded on non-volatile media
- Resourceful – delegates "DAGs" of work to other masters
- Speculative – takes chances and knows how to recover from failure
- Self aware – knows its own capabilities and limitations
- Obedience – manages work according to plan
- Reliable – can mange "large" numbers of work items and resource providers
- Portable – can be deployed "on the fly" to act as a "sub master"

Master should not do …

- Predictions …
- Optimal scheduling …
- Data mining …
- Bidding …
- Forecasting …

The Ethernet Protocol

IEEE 802.3 CSMA/CD - A truly distributed (and very effective) access control protocol to a shared service.

- Client responsible for access control
- Client responsible for error detection
- Client responsible for fairness

Never assume that

what you know

is still true and that

what you ordered

did actually happen.

Every Community can benefit from the services of

Matchmakers!

eBay is a matchmaker

Why? Because ...

.. someone has to bring together community members who have requests for goods and services with members who offer them.

- Both sides are looking for each other
- Both sides have constraints
- Both sides have preferences

Being a Matchmaker

- Symmetric treatment of all parties
- Schema "neutral"
- Matching policies defined by parties
- "Just in time" decisions
- Acts as an "advisor" not "enforcer"
- Can be used for "resource allocation" and "job delegation"

Bringing it all Together


desktop

From

Condor

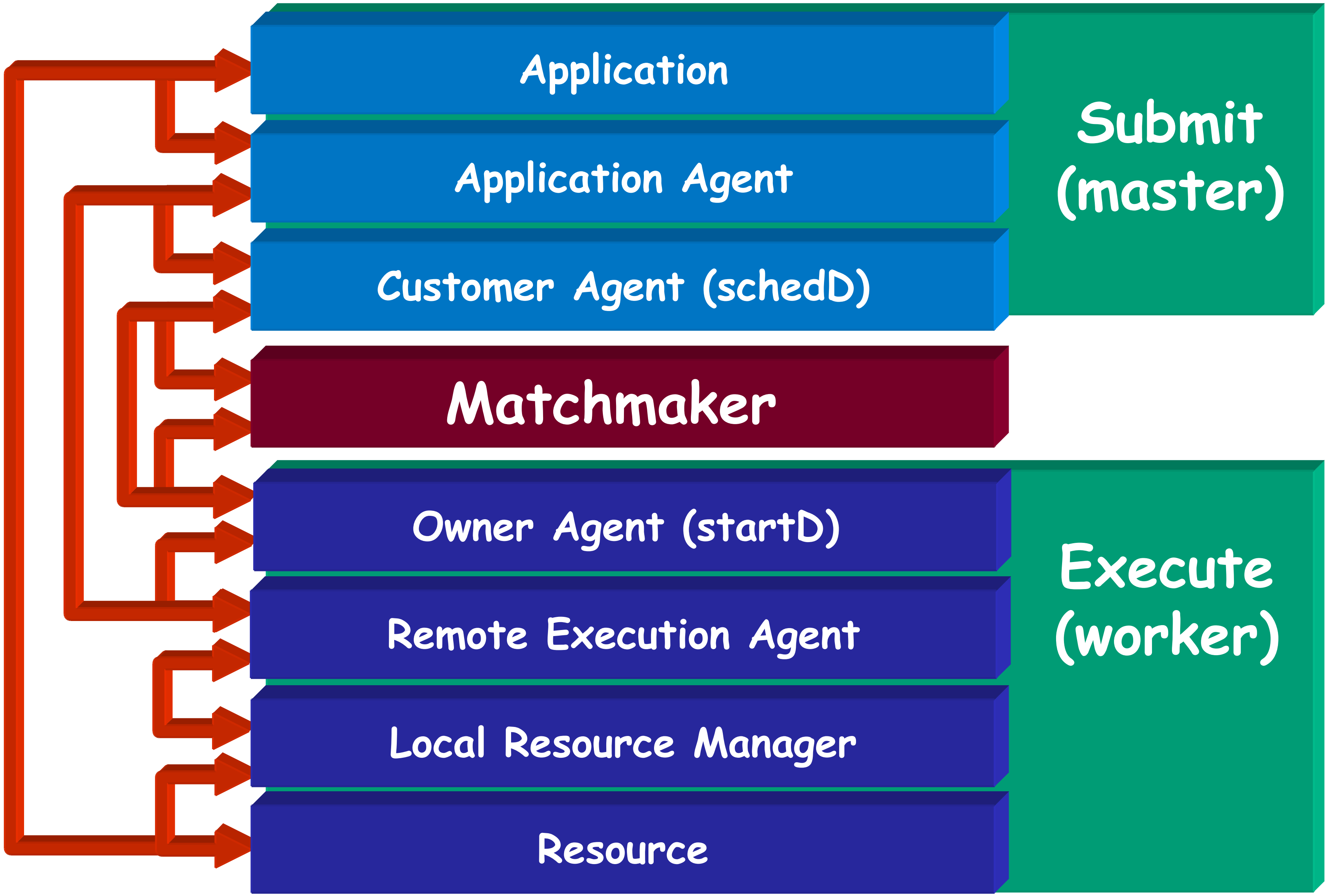
to

Condor-G

to

Condor-C

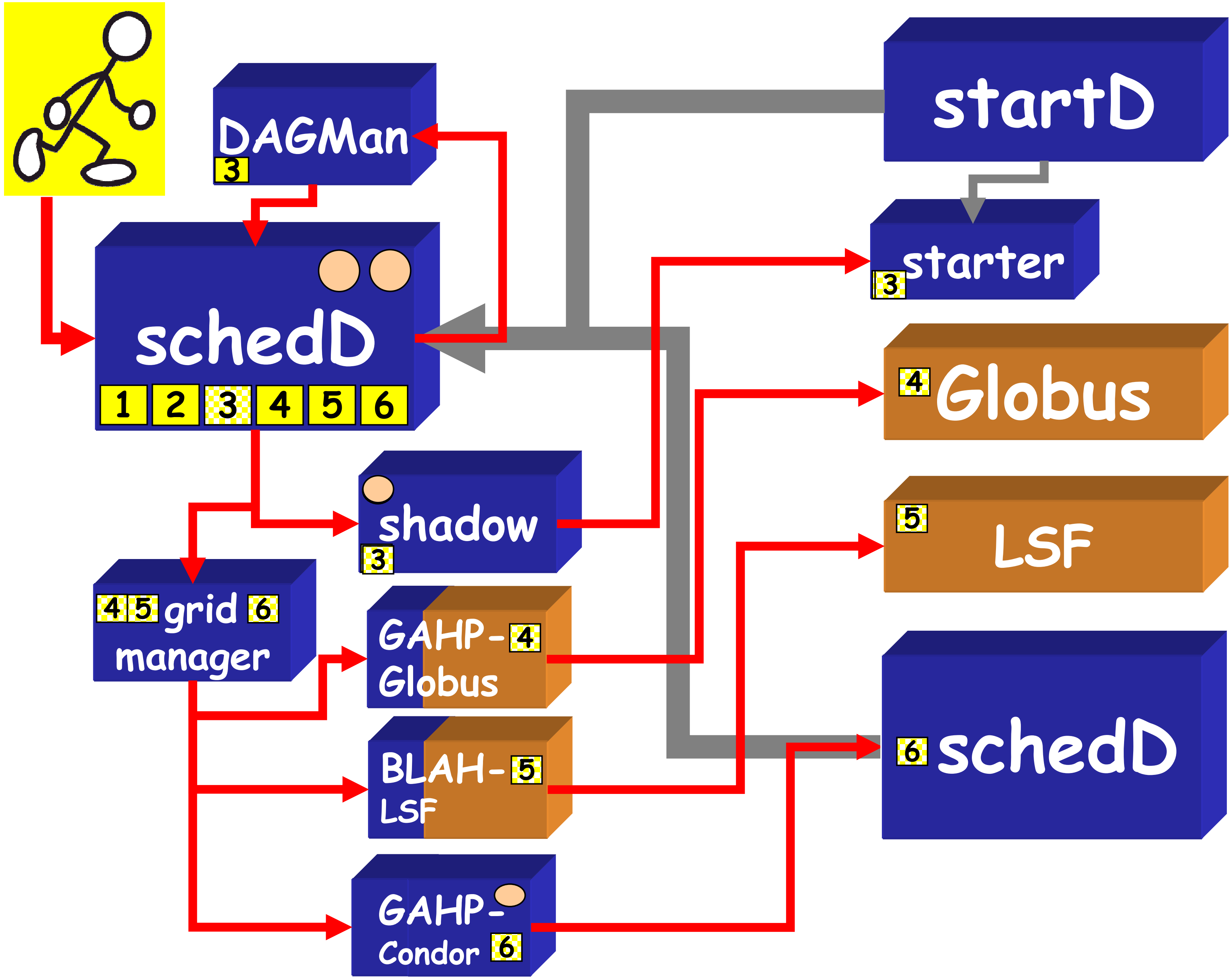
The Layers of Condor

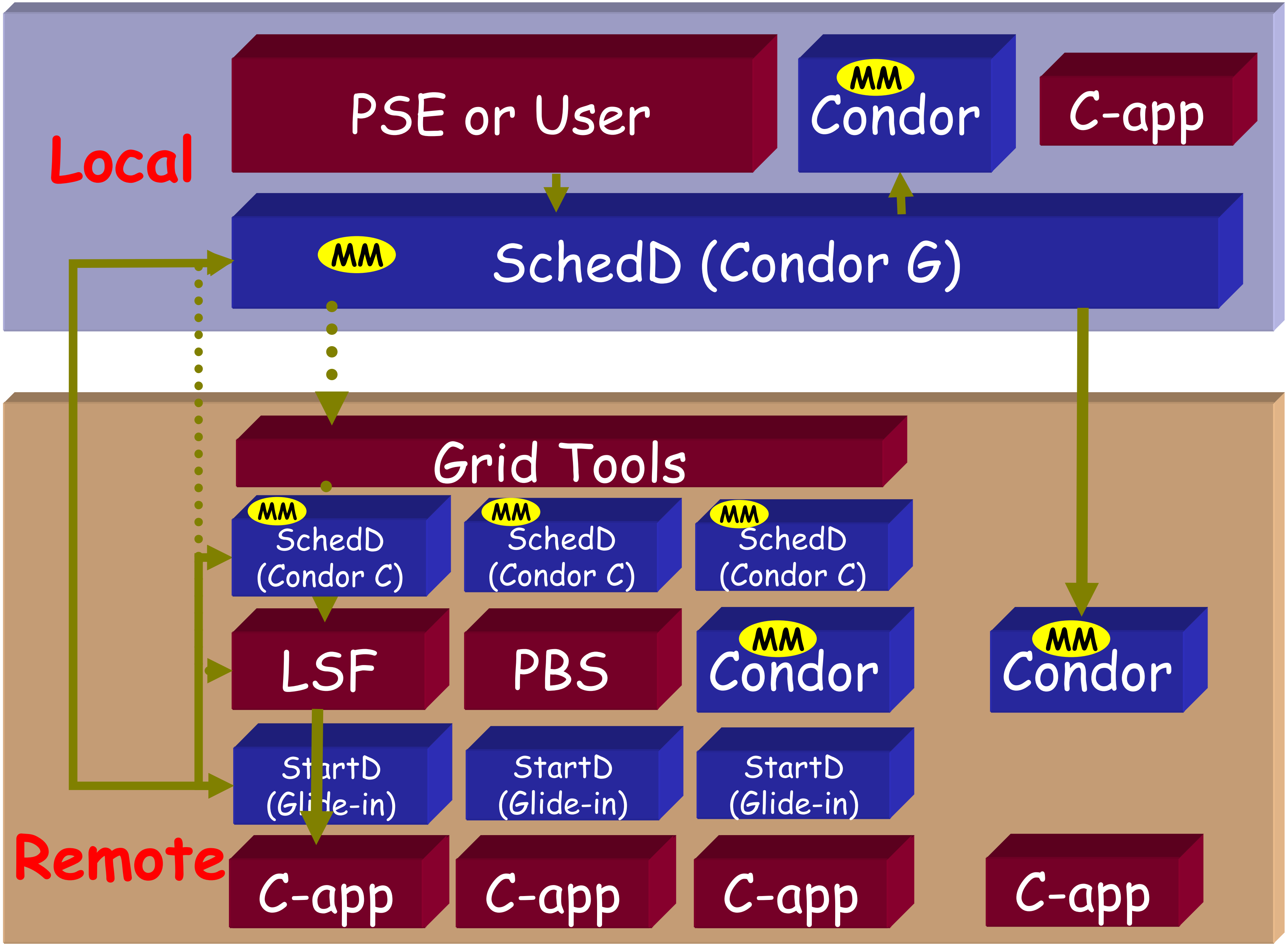


Be matched, claim (+maintain), and then delegate

Job Submission Options

- leave_in_queue = <ClassAd Boolean Expression>
- on_exit_remove = <ClassAd Boolean Expression>
- on_exit_hold = <ClassAd Boolean Expression>
- periodic_remove = <ClassAd Boolean Expression>
- periodic_hold = <ClassAd Boolean Expression>
- periodic_release = <ClassAd Boolean Expression>
- noop_job = <ClassAd Boolean Expression>


DAGMan
3
schedD
1 2 3 4 5 6
shadow
3
4 5 grid 6
manager
GAHP-4
Globus
BLAH-5
LSF
GAHP-
Condor 6
startD
starter
3
4 Globus
5 LSF
6 schedD



Local
PSE or User
MM
Condor
C-app
MM
SchedD (Condor G)
Grid Tools
MM
SchedD
(Condor C)
MM
SchedD
(Condor C)
MM
SchedD
(Condor C)
LSF
PBS
MM
Condor
MM
Condor
StartD
(Glide-in)
StartD
(Glide-in)
StartD
(Glide-in)
Remote
C-app
C-app
C-app
C-app

How can we accommodate
an unbounded
need for computing with
an unbounded
amount of resources?

The words of Koheleth son of David, king in Jerusalem ….

Only that shall happen
Which has happened,
Only that occur
Which has occurred;
There is nothing new
Beneath the sun!

Ecclesiastes Chapter 1 verse 9

Close by storage is

small and **fast**

faraway storage is

big and slow

Many data challenges …

Managing data is a hard problem. Doing it in a distributed environment does not make it easier or simpler:

- Catalogs and metadata
- Access control
- Consistency and coherency
- Revocation and auditing
- Replication/cashing management
- Planning (optimization?)

Almost everything we do requires a

dependable

data placement mechanism

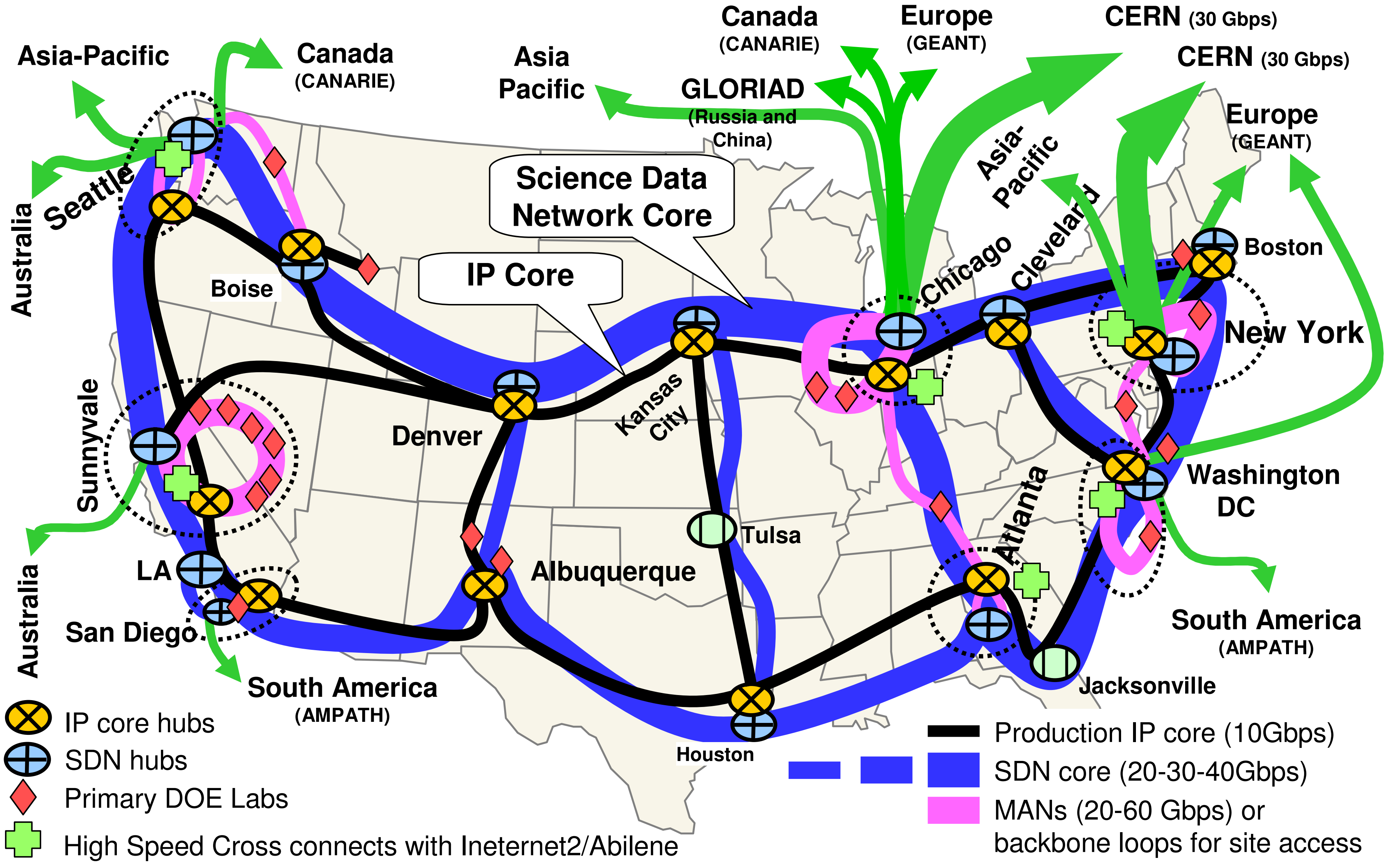
We are making progress …

- The Storage Resource Management (SRM) protocol – management of file copies and support for space reservations
- The Reliable File Transfer (RFT) service – management of large numbers of GridFTP requests
- The File Transfer Service (FTS) – manages file transfer requests and supports the concept of "channels"
- The Planning for Execution in Grids (Pegasus) planner – supports data placement steps in the workflow

High capacity networks are deployed all over the world and almost everyone is concerned about how to allocated their bandwidth. However, is bandwidth the real issue?

ESnet4 Target Configuration

Core networks: 40-50 Gbps in 2009, 160-400 Gbps in 2011-2012



Fiber path is ~ 14,000 miles / 24,000 km

Main trend

The ratio between the size of the organization and the volume (and complexity) of the data/information/knowledge the organization owns/manages/depends on will continue to dramatically increase

- Ownership cost of managed storage capacity goes down
- Data/information/knowledge generated and consumed goes up
- Network capacity goes up
- Distributed computing technology matures and is more widely adopted

Managed Object Placement

Management of storage space and bulk data transfers plays a key role in the end-to-end effectiveness of many scientific applications:

- Object Placement operations must be treated as "first class" tasks and explicitly expressed in the work flow
- Fabric must provide services to manage storage space
- Object Placement schedulers and matchmakers are needed
- Object Placement and computing must be coordinated
- Smooth transition of Compute/Placement interleaving across software layers and granularity of compute tasks and object size
- Error handling and garbage collection

Customer requests:

Place $y = F(x)$ at L!

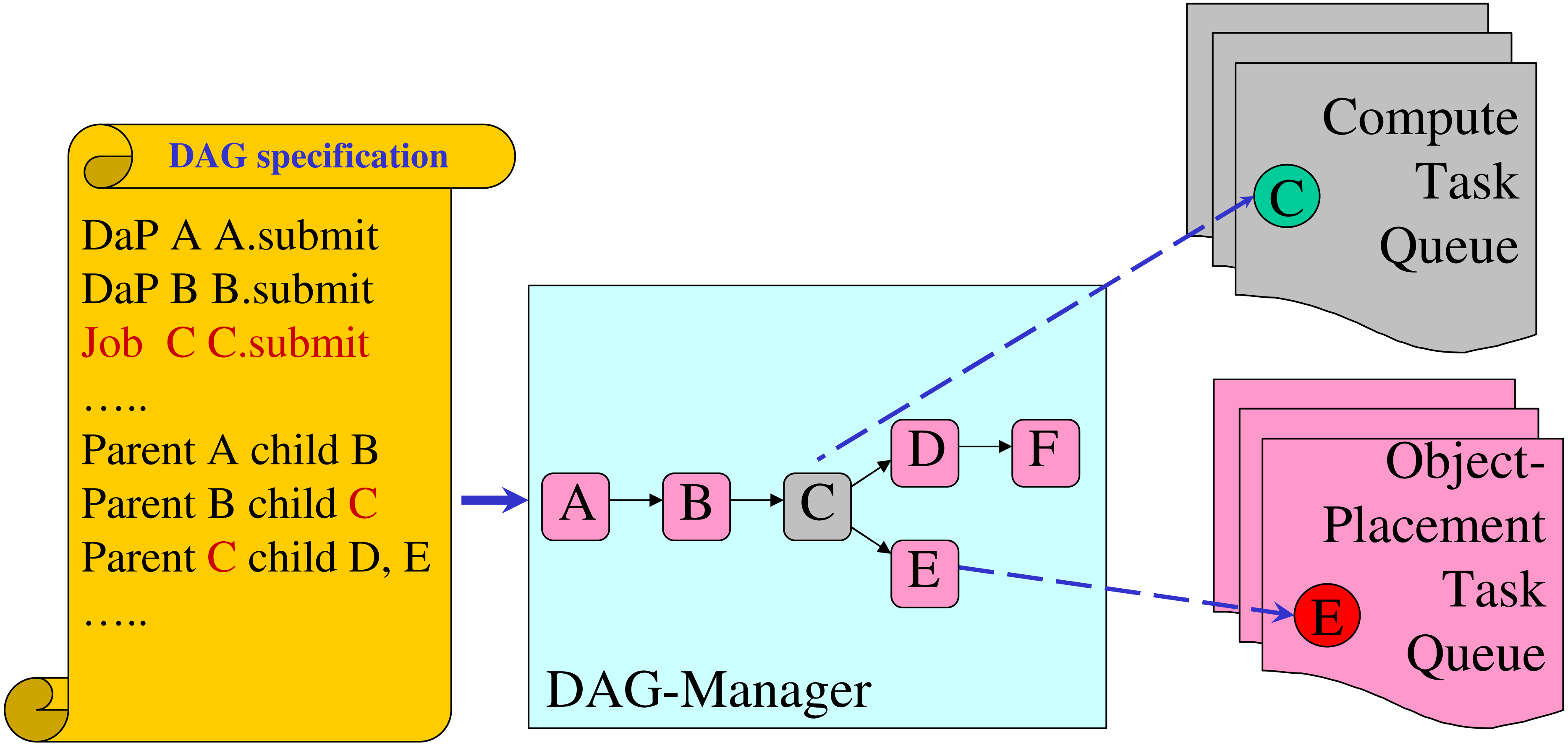
System delivers.

Simple plan for *y=F(x)→L*

1. Allocate (size(x)+size(y)+size(F)) at SE_i
2. Place x from SE_j at SE_i
3. Place F on CE_k
4. Compute F(x) at CE_k
5. Move y from SE_i at L
6. Release allocated space at SE_i

Storage Element (SE); Compute Element (CE)

The Basic Approach*

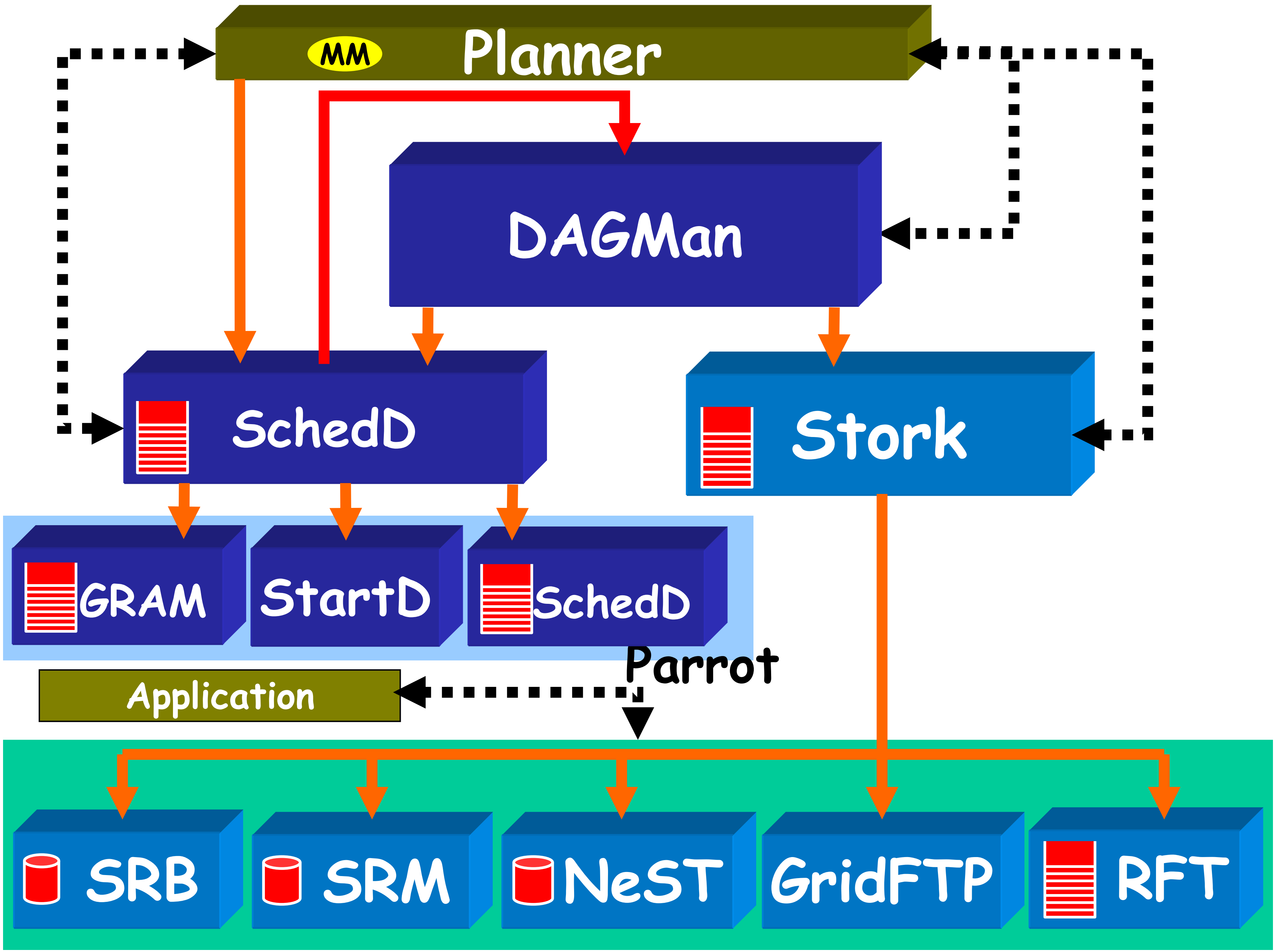


*** DAG – Directed Acyclic Graph**

Stork-A possible solution

A portable, flexible and extensible Object Placement Scheduler.

- Uses ClassAds to capture jobs and policies (just like Condor)
- Supports matchmaking (just like Condor)
- Provides a suite of data transfer jobs that interface with a broad collection of storage systems and protocols and provide end-to-end reliability
- Supports storage allocate/release jobs


MM
Planner
DAGMan
SchedD
Stork
GRAM
StartD
SchedD
Parrot
Application
SRB
SRM
NeST
GridFTP
RFT

Customer requests:

Place y@S *at* L!

System delivers.

Basic step for y@S→L

1. Allocate size(y) at L,
2. Allocate resources (disk bandwidth, memory, CPU, outgoing network bandwidth) on S
3. Allocate resources (disk bandwidth, memory, CPU, incoming network bandwidth) on L
4. Match S and L

Or in other words,

it takes **two** (or more)

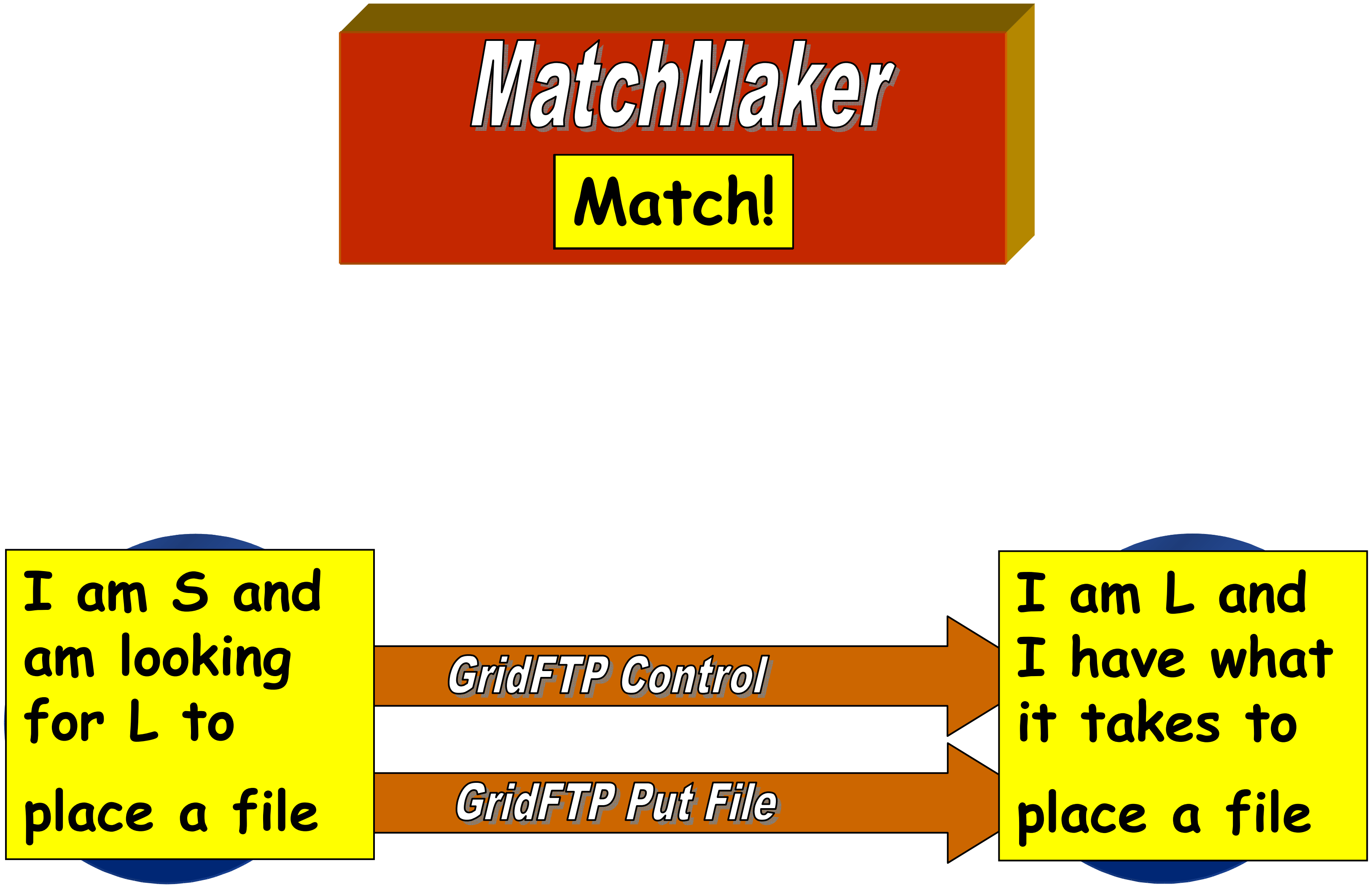
to Tango

(or to place an object)!

When the
“source” plays “nice”
it “asks”

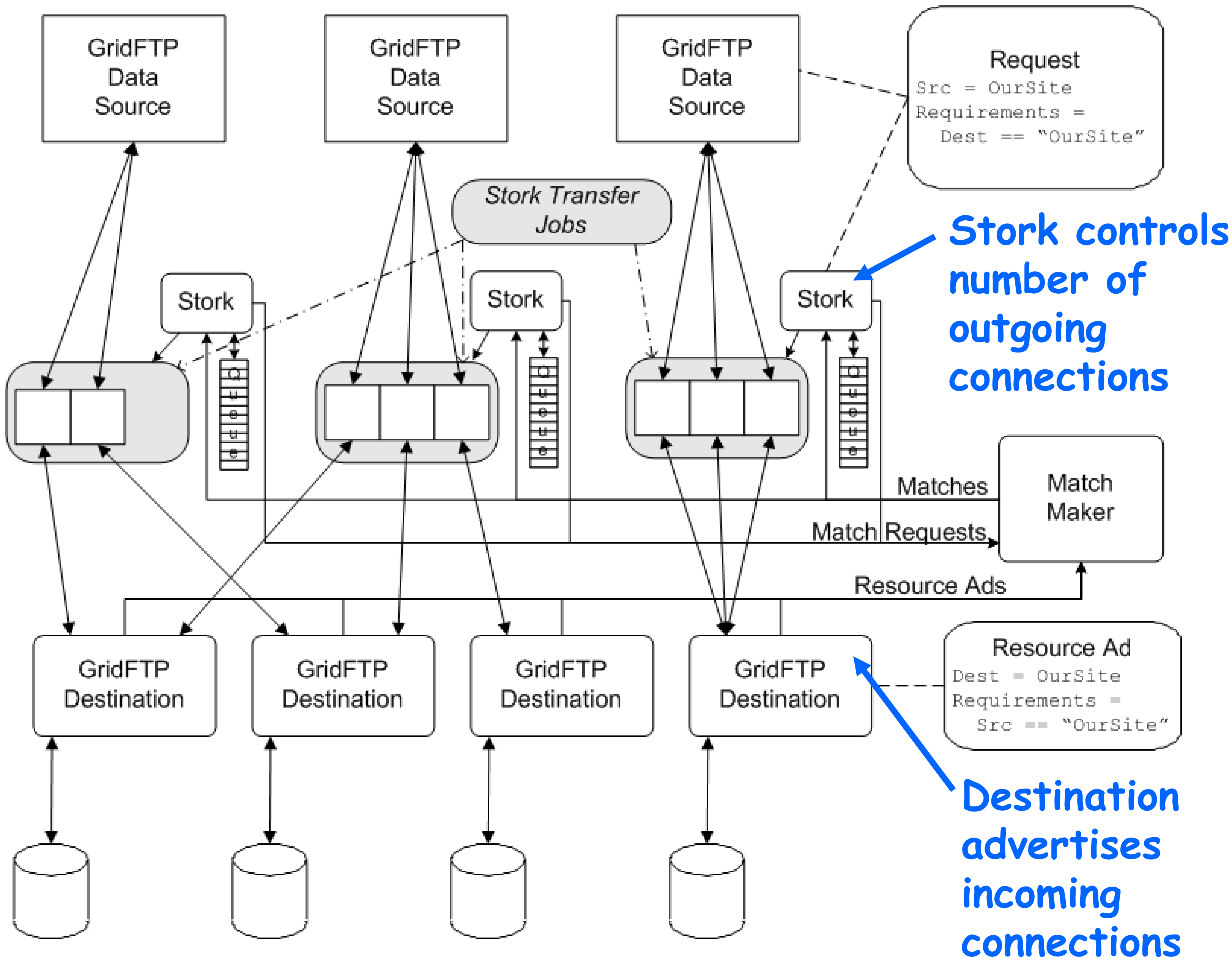
in advance

for permission to
place an object at the
“destination”

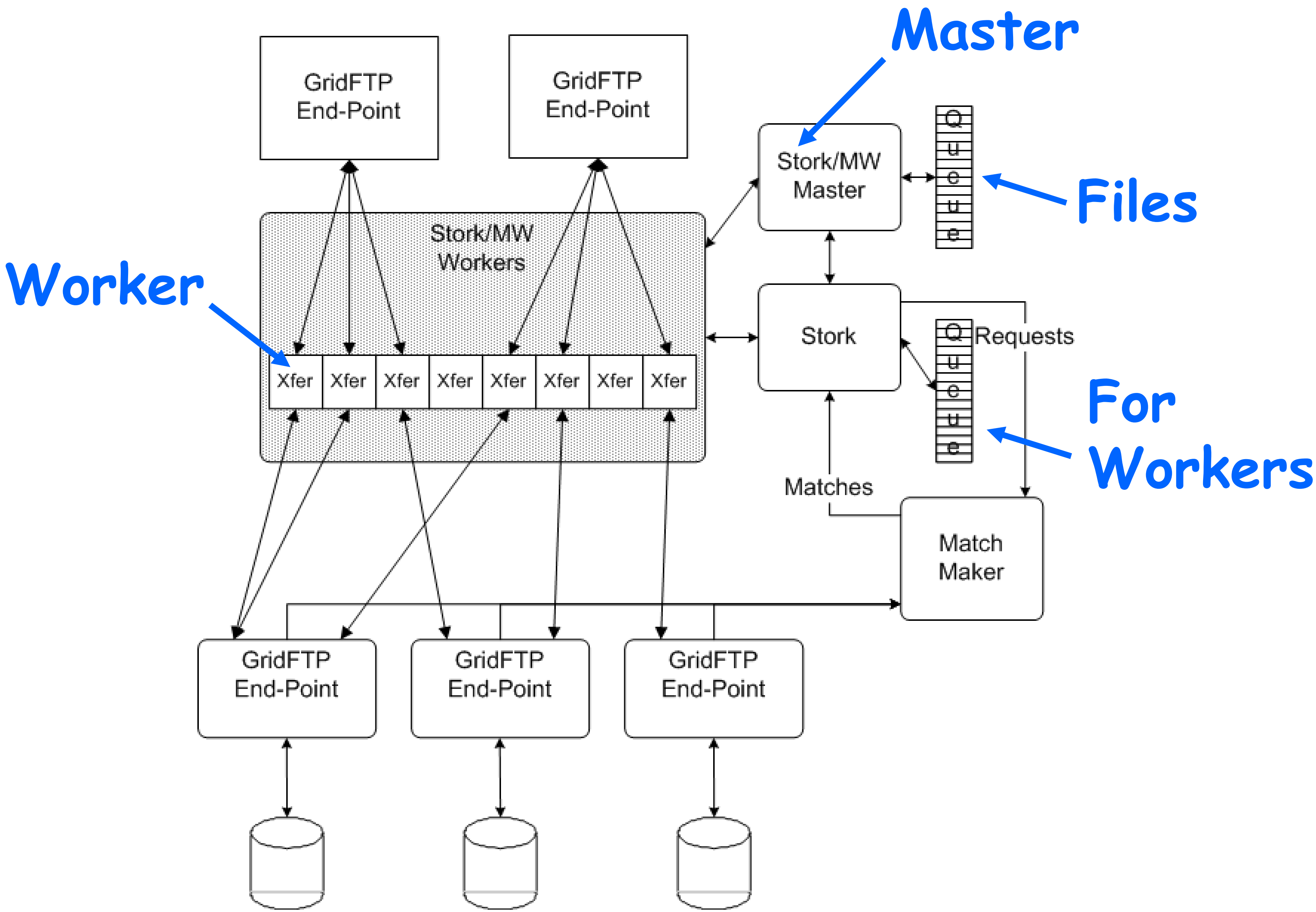

MatchMaker
Match!
I am S and am looking for L to place a file
GridFTP Control
GridFTP Put File
I am L and I have what it takes to place a file

The SC'05 effort

Joint with the Globus GridFTP team


GridFTP Data Source
GridFTP Data Source
GridFTP Data Source
Request
Src = OurSite
Requirements =
Dest == "OurSite"
Stork Transfer Jobs
Stork
Stork
Stork
queue
queue
queue
Stork controls number of outgoing connections
Matches
Match Maker
Match Requests
Resource Ads
GridFTP Destination
GridFTP Destination
GridFTP Destination
GridFTP Destination
Resource Ad
Dest = OurSite
Requirements =
Src == "OurSite"
Destination advertises incoming connections

A Master Worker view of the same effort

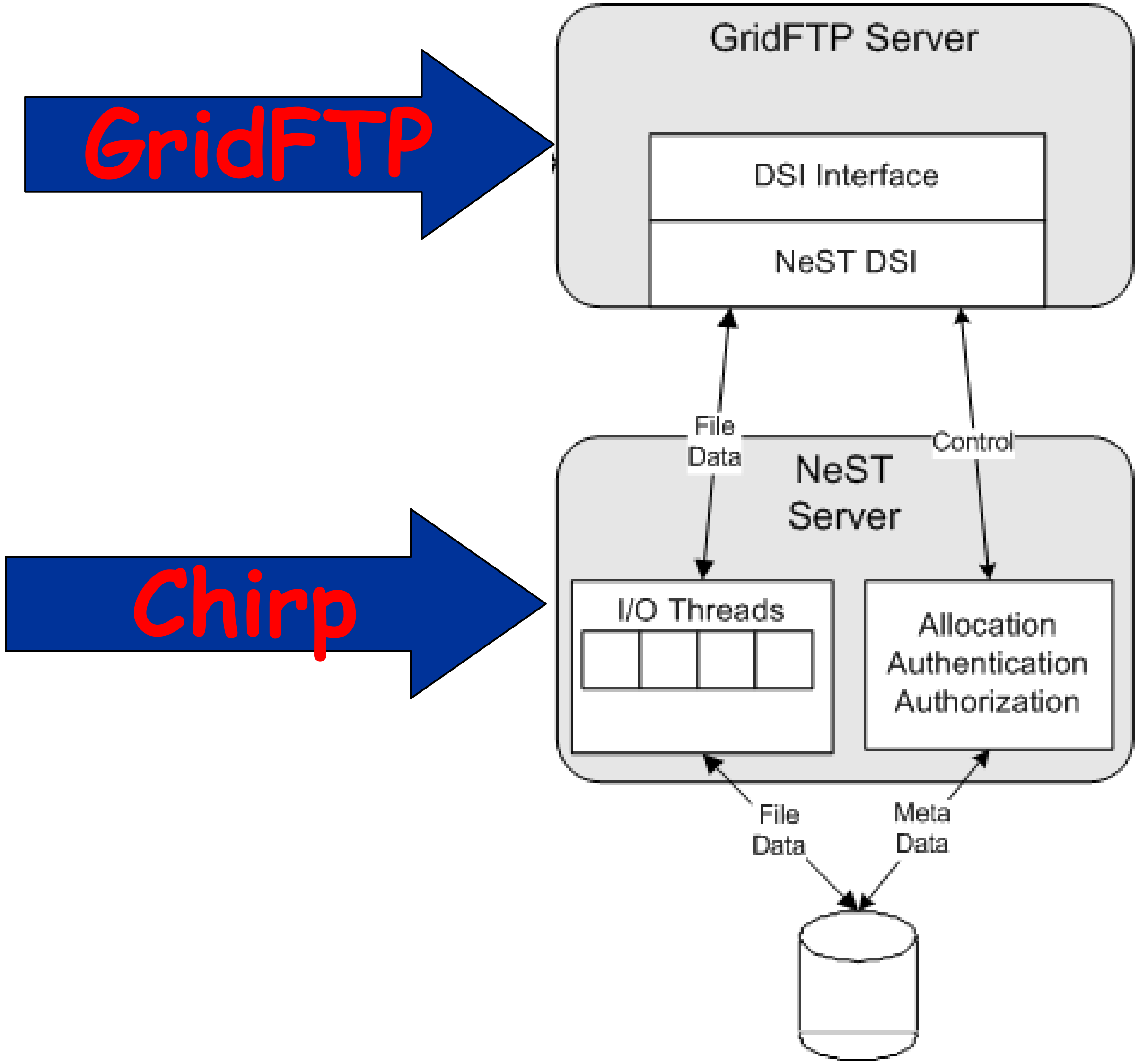

Master
GridFTP End-Point
GridFTP End-Point
Stork/MW Master
Queue
Files
Stork/MW Workers
Worker
Xfer
Xfer
Xfer
Xfer
Xfer
Xfer
Xfer
Xfer
Stork
Queue
Requests
For Workers
Matches
Match Maker
GridFTP End-Point
GridFTP End-Point
GridFTP End-Point

When the "source" does not play "nice", destination must **protect itself**

NeST

Manages storage space and connections for a GridFTP server with commands like:

* ADD_NEST_USER
* ADD_USER_TO_LOT
* ATTACH_LOT_TO_FILE
* TERMINATE_LOT


GridFTP
Chirp
GridFTP Server
DSI Interface
NeST DSI
File
Data
Control
NeST
Server
I/O Threads
Allocation
Authentication
Authorization
File
Data
Meta
Data

How can we accommodate an **unbounded** amount of data with an **unbounded** amount of storage and network bandwidth?