



Petaflops and the Grid

Andrew Grimshaw
University of Virginia
Petaflops II
February 18, 1999



Overview

- The Opportunity
- Challenges
- Current state of the art
- Take-away

The first petaflops computer will be a metasystem — admittedly it will not have the bandwidth and latency required for many applications.

Unless it is a grid-in-a-boxTM



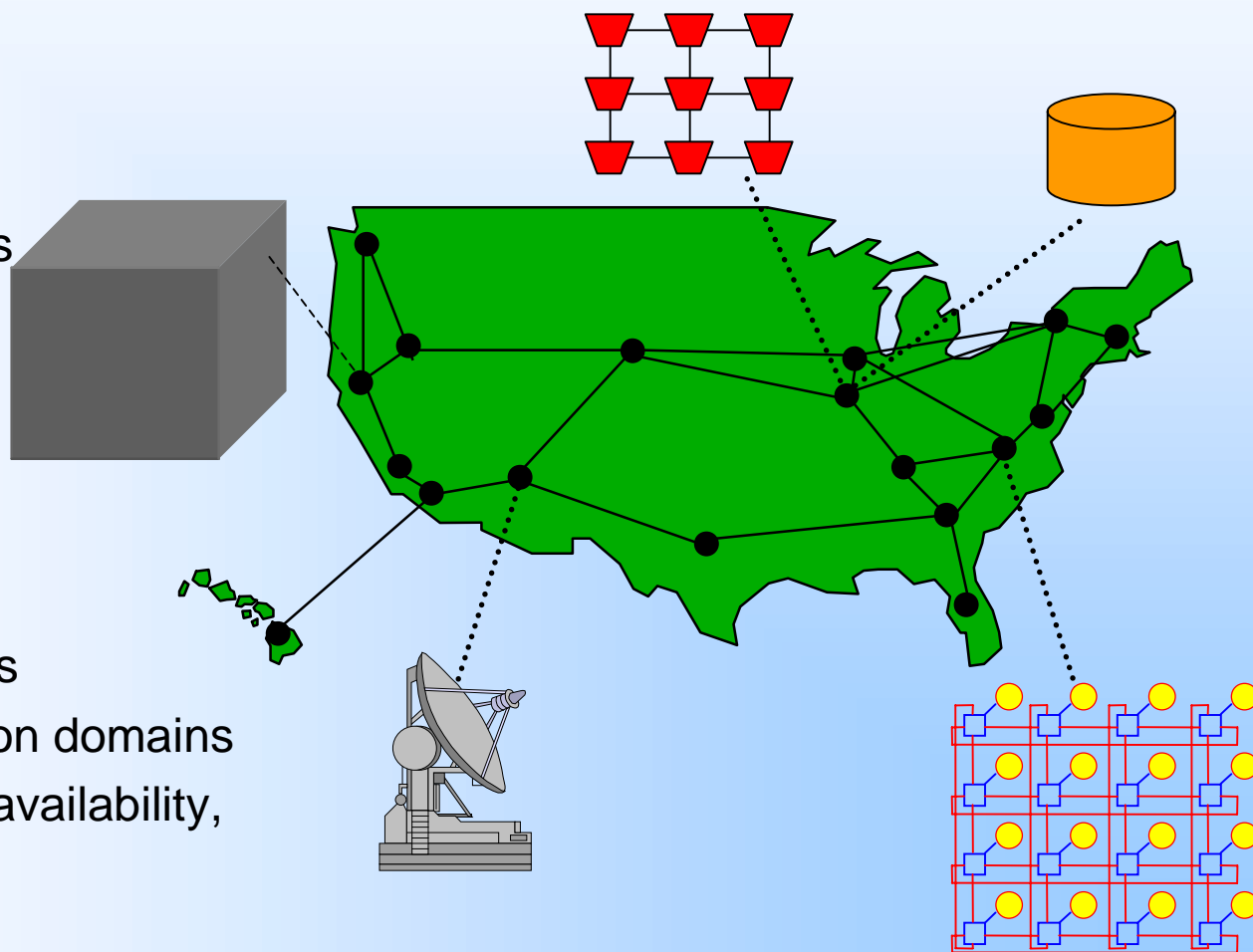
The Opportunity



Emerging Environment

The petaflops machine will not exist in a vacuum

- Resources abound
 - Computer systems
 - High-speed networks
 - People
 - Data
 - Special devices
- Complex environment
 - Disjoint file systems
 - Disjoint name spaces
 - Multiple administration domains
 - Unpredictable load, availability, failures
 - Security problems



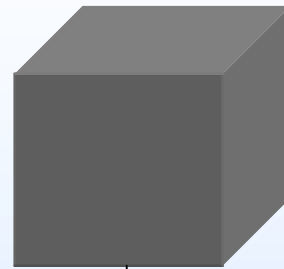
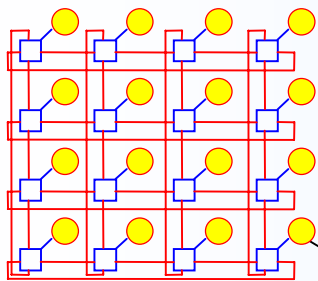


Scenarios



Collaboration

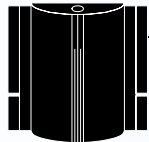
Model running on workstation cluster at UCLA generates preconditions.



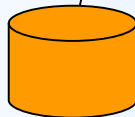
The "Box"



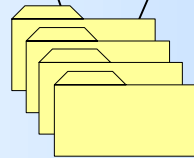
Scientists at MIT execute simulation using data stored on the HPSS at SDSC.



Simulation uses/stores results in HPSS system. Post-processing done on local system



Colleague at NAVO "watches".

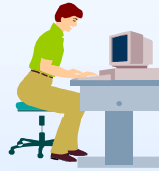


Reusable simulation components are made available by developers at NCSA and LANL

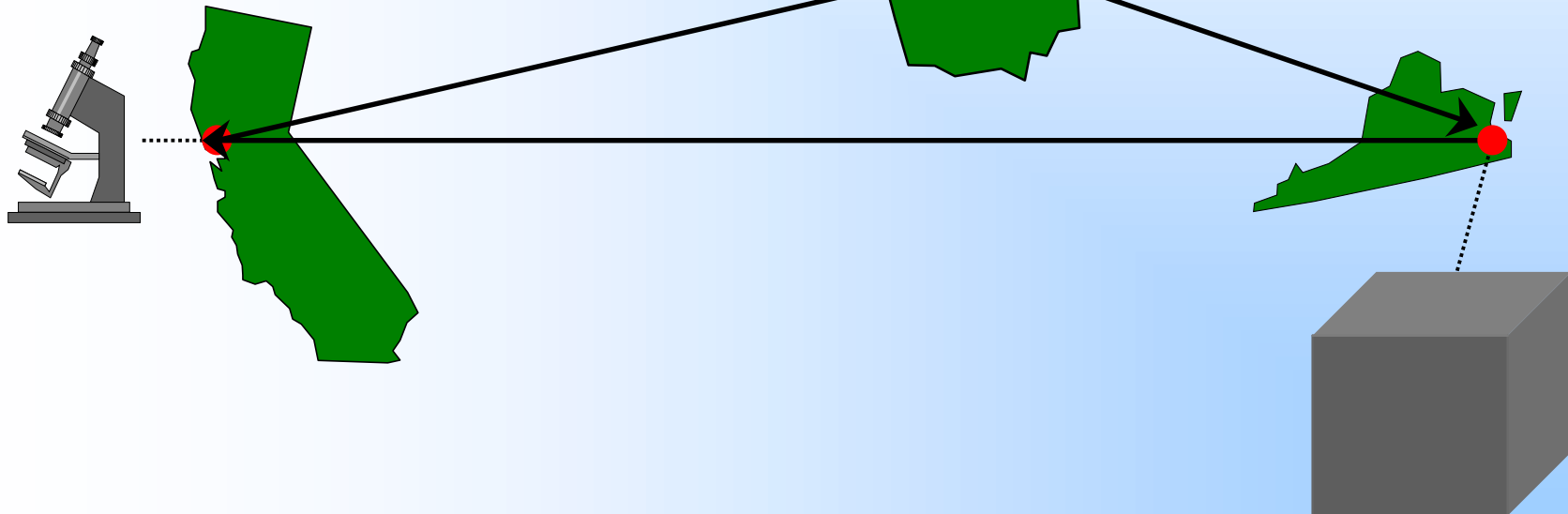


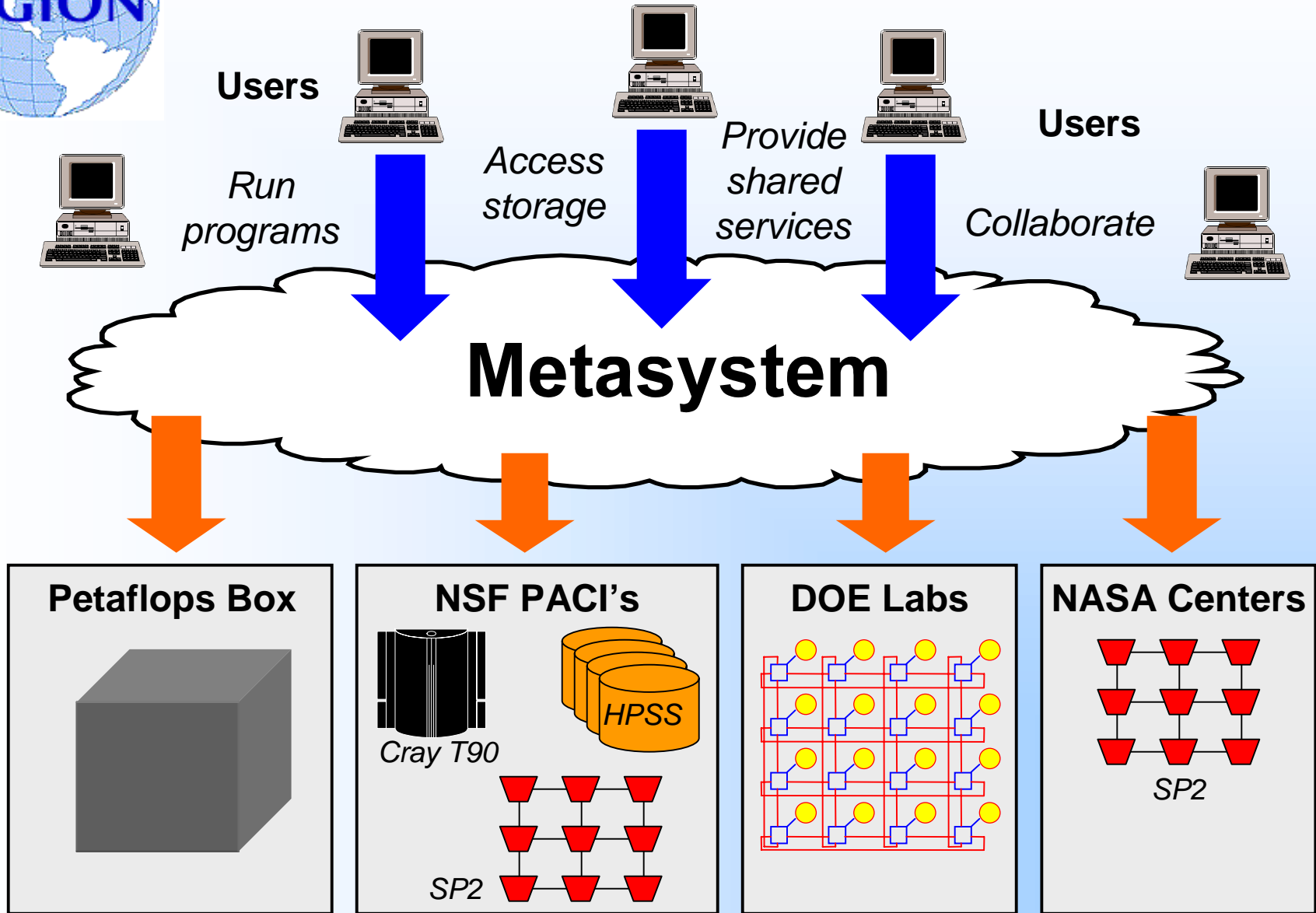
Real-time Device Interactions

*Wind Tunnel at
NASA Ames*



*User at
Lewis*







Challenges

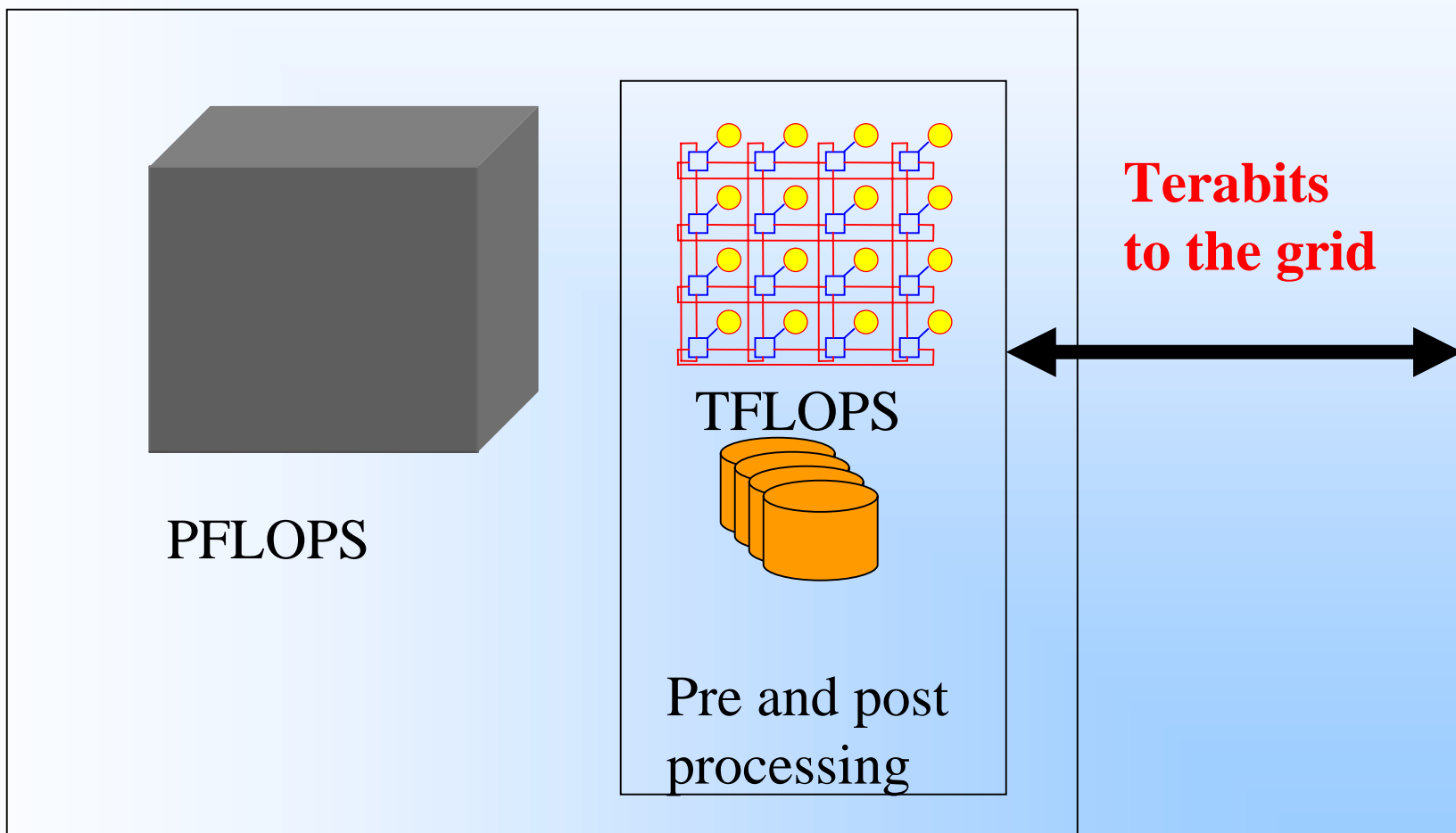


Challenges/requirements

- Transparent access to resources — data, devices, compute
- Security and access control
- Site autonomy (no OS mods)
- User expectations
- Scheduling and resource management
 - co-scheduling
- Fault-tolerance
- I/O
- Site bandwidth — lots of it



Site Requirements





Resource Management

- Co-scheduling of resources necessary
 - No good to get the petaflop machine if you can't access the data/device/whatever when needed
 - Need resource reservation mechanisms and policies that facilitate acquiring all of the resources needed to solve the problem — including the petaflops machine



Fault-tolerance

- In a grid there are many things that can fail — failure will be the norm not the exception
 - We cannot allow external failures to abort a petaflops-class job in the middle
- We also must deal with failure inside the machine
 - Fault-tolerance has not been a priority to date — we'll pay for that as time moves on



The Java-Generation is coming

- The next generation of scientists and engineers that will use the petaflops computer have much higher expectations
 - software that
 - works
 - is easy to use — they're not used to thinking at the architecture level
 - is composable
- Windows — love it or ...else



The State of the Art



Metacomputing today

- A number of agencies have committed themselves to metacomputing/grids
 - NSF PACI
 - NASA IPG
 - DOE DISCOM2
 - DOD HPCMOD
- These are precursors
- Grids forum



Ongoing projects

- Globus (Foster@ ANL, Kesselman@ISI)
 - Sum of services architecture
- Legion (Grimshaw@Virginia)
 - Object-based architecture
- Others
 - Harness - Geist, Sunderam, Dongarra
- The Book - "Building the Grid" Morgan Kaufman 98



Globus and Legion

- Projects share many common goals:
 - Metacomputing (or the “Grid”)
 - Middleware for wide-area systems
 - Heterogeneous resource sets
 - Disjoint administrative domains
 - High-performance, large-scale applications



Some non-shared goals

- Legion
 - Shared collaborative environment
 - Both HPC applications and distributed applications
 - Create a meta-operating system



Many “Similar” Features

- Resource Management Support
- Message-passing libraries
 - e.g., MPI
- Distributed I/O Facilities
 - Globus GASS vs. Legion FS
- Security Infrastructure
- Fault monitoring and response



Basic Differences

- Architecture
 - Legion and Globus were designed and built under very different architectural philosophies
- Features
 - The systems provide qualitatively different interfaces and abstractions



Globus

- The “toolkit” approach
 - Provide services as separate libraries
 - e.g., Nexus, GASS, LDAP
 - Pros:
 - Decoupled architecture
 - easy to add new services into the mix
 - Low buy-in: use only what you like!
 - Cons:
 - No unifying abstractions
 - very complex environment to learn in full
 - composition of services difficult as number of services grows



Legion

- Object-oriented environment approach
 - Provide a virtual Object-oriented meta-OS
 - Pros:
 - A small set of simple, composable abstractions
 - Can be composed to construct new services
 - Less to learn — same tools can be used to control many services
 - Object subclassing
 - Replace/augment/control the implementations of local services with ease: **true autonomy**
 - Cons:
 - Perceived high buy-in cost
 - Have to run within the “whole” environment
 - Reuse of existing “non-Legion” services is harder



Where are they now?

- These projects are not vapor-ware — they are running today
- Globus is going into pre-production at NASA, NPACI, and NCSA
- Legion is in test at NPACI, NASA, DOD



is the way to go!

Take-away message

- The petaflops-in-a-box machine will not exist in a vacuum
- The grids are coming —b they'll be in place when the petaflops machine arrives
- The first petaflops machine will be a metasystem