Concepts and Architecture of Grid Computing

Advanced Topics Spring 2008

Prof. Robert van Engelen

Overview

- Grid users: who are they?
- Concept of the Grid
- Challenges for the Grid
- Evolution of Grid systems
- Model of the Grid architecture
- Condor-G: Globus and Condor
- Applications

Grid Users

- Computational/experimental scientists and engineers
  - Control remote instruments
  - Access to data repositories and (super)computers
  - Visualization and data analysis
- Industry
  - Link people (video conferencing & remote collaboration systems)
  - Share resources
  - Problem solving
- Government
  - Public policy making based on collaborative information sharing
  - Disaster response using information gathering and decision making
- Education
  - Virtual classrooms (one-way video conferencing)
  - Remote demonstrations and remote access to collections

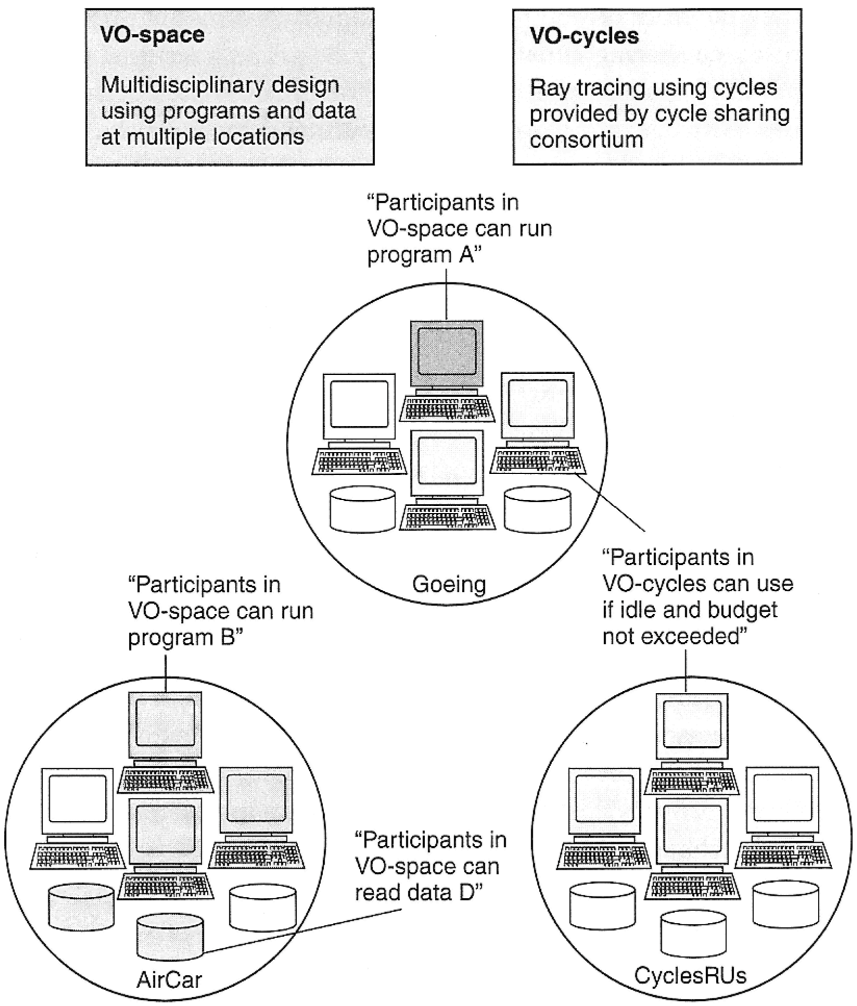
Scientific Users of the Grid

- Scientists and engineers may need the Grid for…
  - Data intensive applications
    - Collider experiments in particle physics produce petabytes per year
    - Astronomy digital sky surveys, e.g. the virtual observatory project
    - Example data grids: NSF GriPhyN, DOE PPDG, EU DataGrid
  - Imaging
    - Managing collections of medical images: MRI, CT scans, X-rays, …
    - Comparing images and metadata for epidemiological studies
  - Scientific simulations
    - Using supercomputers as scientific instruments
    - Example simulations: Japanese earth simulator, DOE nuclear bomb simulations, computational chemistry, life sciences, …
  - Remote access to instruments
    - Example: network for earthquake engineering simulation

Concept of the Grid

- A common theme underlying these usage modalities is a need for "*coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations*"
- What is a *Virtual Organization* (VO)?
  - A group of individuals or institutions defined by a set of sharing rules
- The Grid concept
  - Grid = a technology that manages two opposing forces: sharing & trust
  - Sharing: coordinated resource sharing in a VO between participants with varying degrees of prior relationships and trust
  - Trust: trust within a VO is precisely defined by the sharing rules governed by security and access policies
- Limited trust means limited sharing
- No trust means no sharing!
- Why trust at all?
  - There is a common goal, could be monetary gain
  - Isolation of shared resources from critical infrastructure, e.g. sandboxing

Resource Sharing Relationships



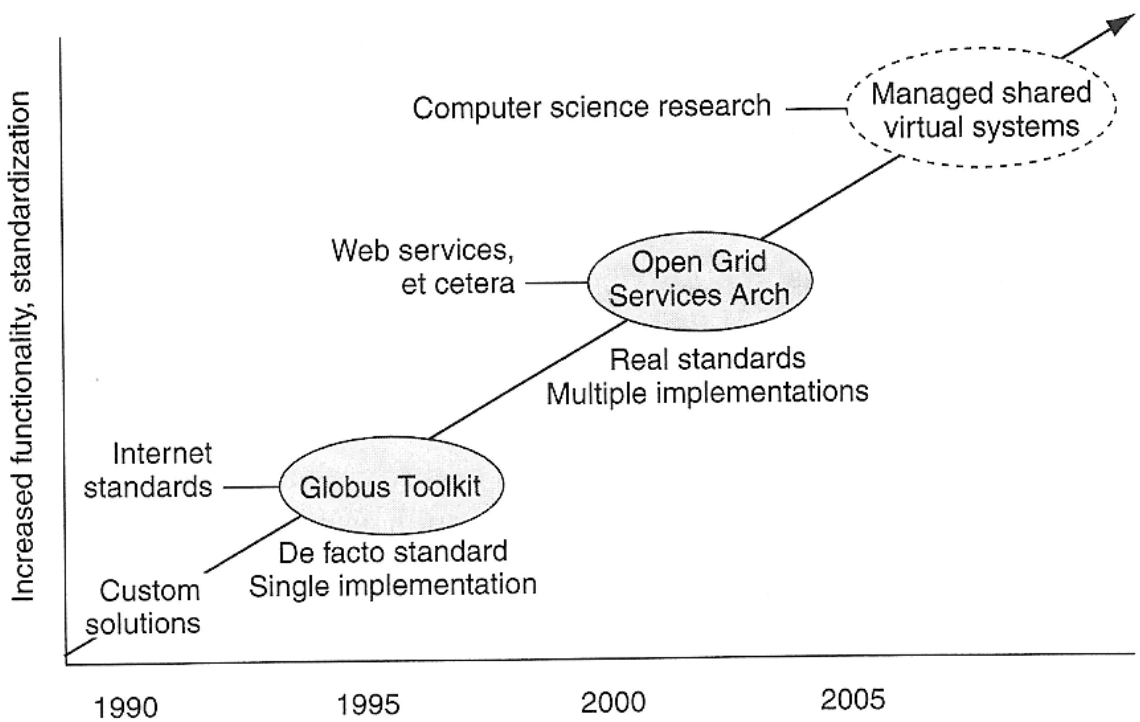
- Physical organizations
  - AirCar, Goeing, CyclesRUs
- Two virtual organizations
  - VO-Space: AirCar and Going collaborate on space project
  - VO-Cycles: ray tracing cycle sharing consortium formed by Goeing and CyclesRUs
- Resource sharing within a VO
  - Is conditional: policies for when, where, and what
    - "can run programs A or B"
    - "can read data D"
    - "can use idle machines if budget not exceeded"
  - Is secure: pooling of resources that are certified as "secure"

Challenges

- Trust (security is built on trusted parties or trusted third-party CA)
  - Problem: how to trust VO members and its agents (autonomous apps)
  - Solution: proxy credentials provided by a CA in public key infrastructure
- Sharing of applications and data
  - Problem: incompatible machines and OS, need to limit access
  - Solution: virtualization, Grid resource allocation policies
- Communication of Grid policies and metadata: Grid interoperability
  - Problem: incompatible protocols
  - Solution: XML-based protocols and open standards
- Reliability and robustness (a non-functional requirement)
  - Problem: Grid-based systems can be brittle (network connections)
  - Solution: two-phase commit, transaction-based protocols
- Quality of service (QoS) (a non-functional requirement)
  - Problem: need end-to-end resource management, transactions
  - Solution: budgeting of cycles, bandwidth, and storage capacity

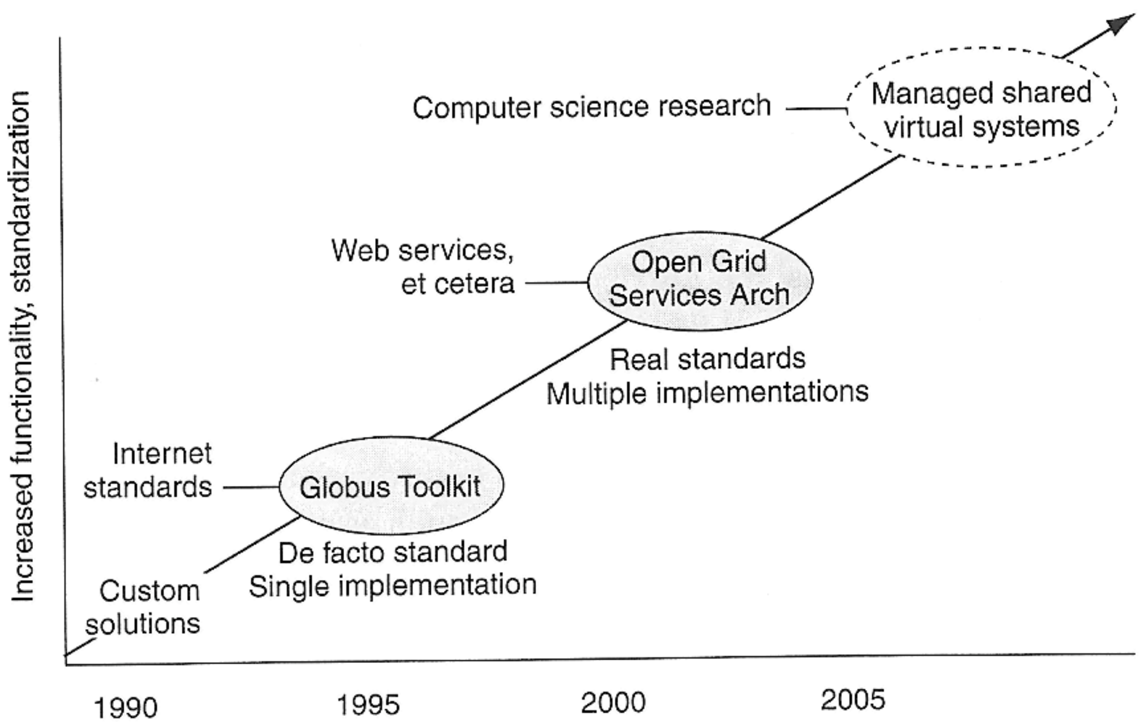
Evolution of Grid Technologies

- Custom solutions (early 90s)
  - "Metacomputing" explorative work
  - Applications built directly on Internet protocols (TCP/IP)
  - Limited functionality, security, scalability, and robustness



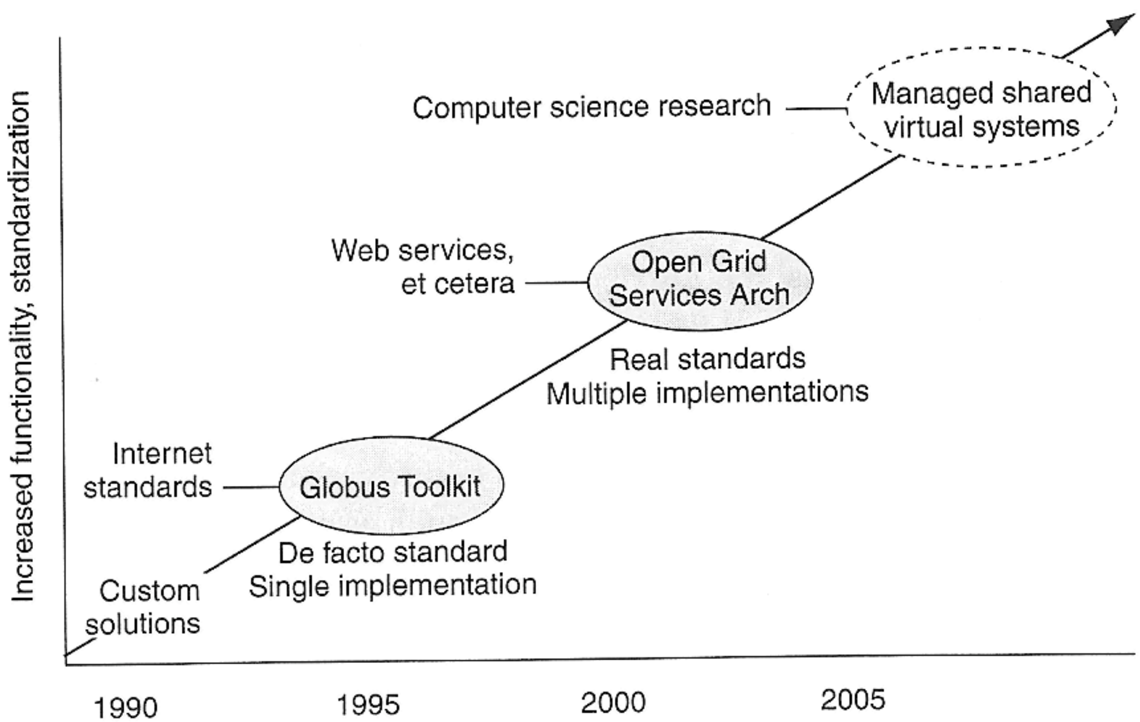
Evolution of Grid Technologies

- Globus toolkit version 2 (GT2) (1997)
  - Popular open-source collection of tools for secure sharing
  - Defined and implemented protocols, APIs, and services
  - De-facto Grid standards



Evolution of Grid Technologies

- Open Grid Services Architecture (OGSA) (2002)
  - Community standard with multiple implementations
    - Globus GT3 implementation
  - Service-oriented architecture based on XML Web services

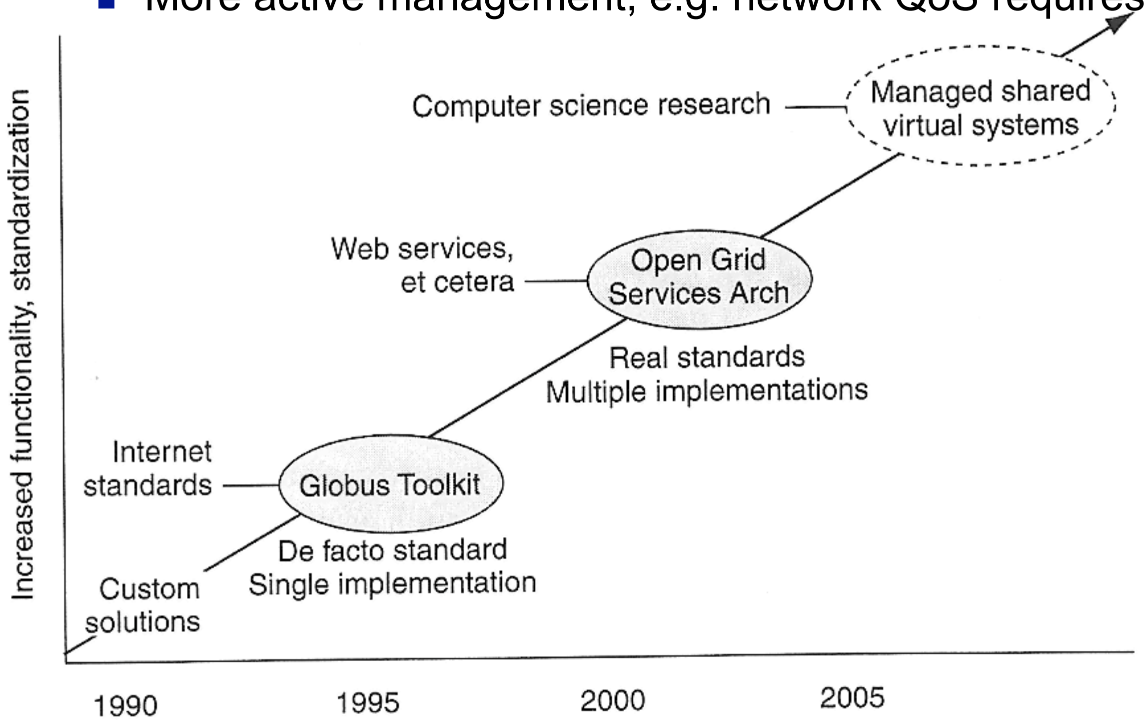


Open Grid Services Architecture

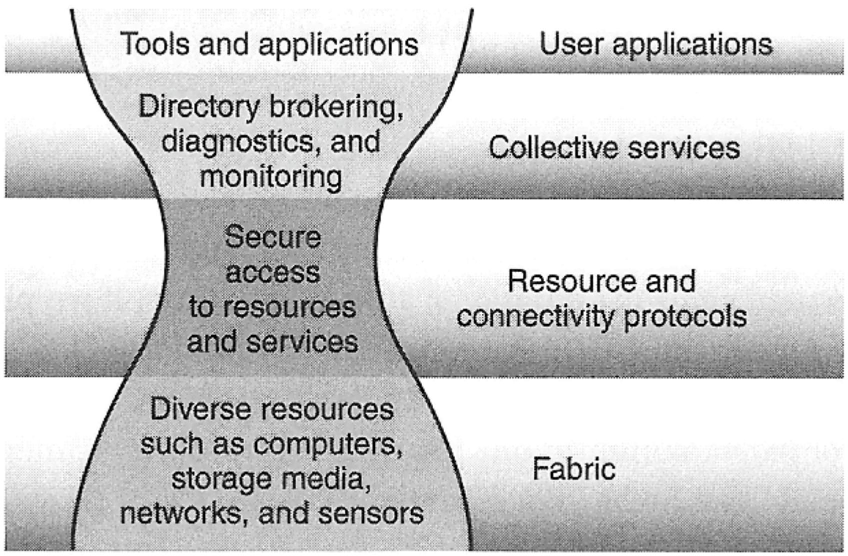
- OGSA (GT3) is a redesign of Globus GT2
- Factoring of component behaviors
  - Problem: current tools combine too many aspects redundantly
  - OGSA identifies basic set of essential services
- Service orientation
  - Services instead of applications
  - A service has a well-defined web interface (like an HTTP server)
- Aligned with XML web services
  - XML-based message protocols
- OGSA's idea was good, but its use is not successful
  - Need simpler, more basic service protocols instead of OGSA-specific extensions of the web service description language (WSDL)
- Globus GT4 uses WS-RF (resource framework) a set of messaging-level protocols for resource properties, lifetime management, …

Evolution of Grid Technologies

- Managed, shared virtual systems (?)
  - Improved technology
    - Virtualization to make Grid more seamless
  - Improved quality of service
    - More active management, e.g. network QoS requires WAN-level support



Model of the Grid Architecture



- The "hour-glass" model of the Grid architecture
  - Thin center: few standards
  - Wide top: many high-level behaviors can be mapped
  - Wide bottom: many underlying technologies and systems
- Role of layers:
  - *Fabric*: interfaces local control
  - *Connectivity*: secure communications
  - *Resource*: sharing a single resource
  - *Collective*: coordinated sharing of multiple resources
  - *Applications*: you decide…

Fabric Layer

- Need to manage a resource
  - Reserve usage in advance
  - Allocate space (e.g. storage)
- Grid fabric layer provides standardized access to local resource-specific operations
- Software is provided to discover
  - Computers (OS version, hardware config, usage load)
  - Storage systems
  - Networks
- Globus General-purpose Architecture for Reservation and Allocation (GARA)

Connectivity Layer

- Need secure connectivity to resources
- Assumes trust is based on user, not service providers
- Use public key infrastructure (PKI)
  - User is recognized by a Certificate Authority (CA) (within Grid)
  - Single sign-on: allow users to authenticate only once
  - Delegation: create proxy credentials to allow services/agents to act on a user's behalf
- Integrate and obey local security policies in global view
  - Unix file permissions, ACLs, …
- Globus Security Infrastructure (GSI)
  - Standardized mechanism for proxy credential creation and mapping to local access authentication scheme (logins)
  - Based on generic services security (GSS) API, which allows applications to perform these security operations autonomously

Resource Layer

- Need access to resources
  - Invoke remote computation
  - Monitor computation
  - Discover resources
  - Perform data transport (e.g. to and from computations)
- Globus Grid Resource Allocation and Management (GRAM)
  - Job manager and reporter
- Globus Monitoring and Discovery Service (MDS-2)
  - Registry of resources and monitoring of resource usage stats
- Globus GridFTP
  - Good old FTP but faster (multiple streams), integrated security
  - Partial file access

Collective Layer

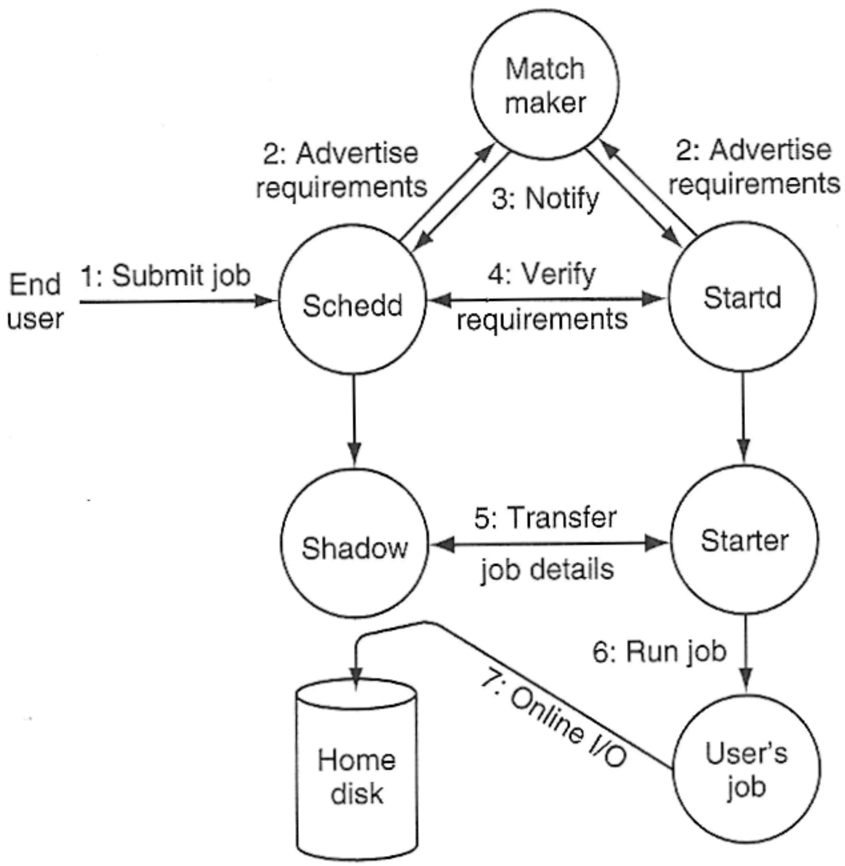
- Need to coordinate sharing of resources
  - Directory services
  - Coallocation, scheduling, brokering services
  - Monitoring and diagnostic services
  - Data replication services
- Tools
  - Workflow systems
  - Collaboratory services
- Globus DUROC resource coallocation library

Resource Allocation via Description Languages

- Condor classified advertisements (ClassAds) describe job and requirements
  - machine architecture
  - command or executable starting application
  - constraints on required resources
  - executable size
  - required memory
  - operating system and release
  - owner
  - rank specifying preferred resources
- Globus RSL uses LDAP-like notation

```
[
 Type = "job";
 Owner = "Myself";
 Executable = "Myimage";
 Rank = other.PhysicalMemory > 256 && other.Mips > 20 || other.Kflops > 2000;
 Imagesize = 275M;
 Constraint = other.Virtual > self.Imagesize;
]
```

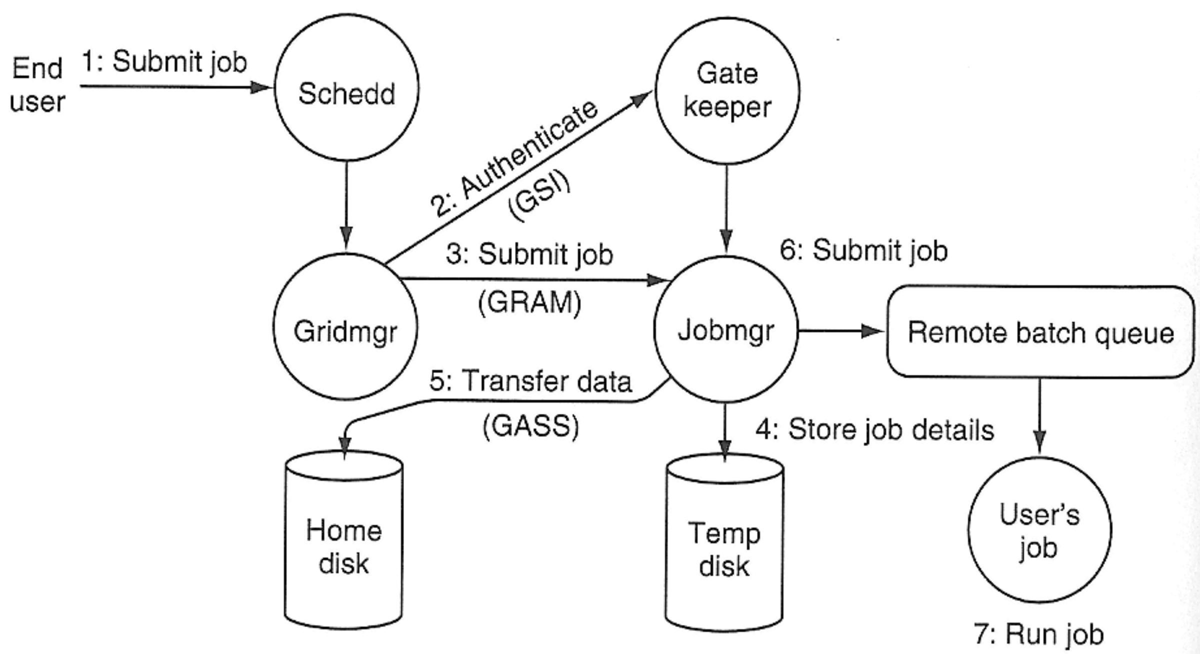
Resource Allocation and Job Scheduling



Condor's 7 steps to run a Condor job

- Resource owner states policy to manage requestors and allocate resources according to rules
  - user access by group via a group authorization policy
  - predefined time slot when job has right to run
  - maximum CPU amount allowed for job
  - maximum disk amount available to group
- Job scheduling and match making based on rules
  - Condor schedd
  - Globus GRAM for RSL

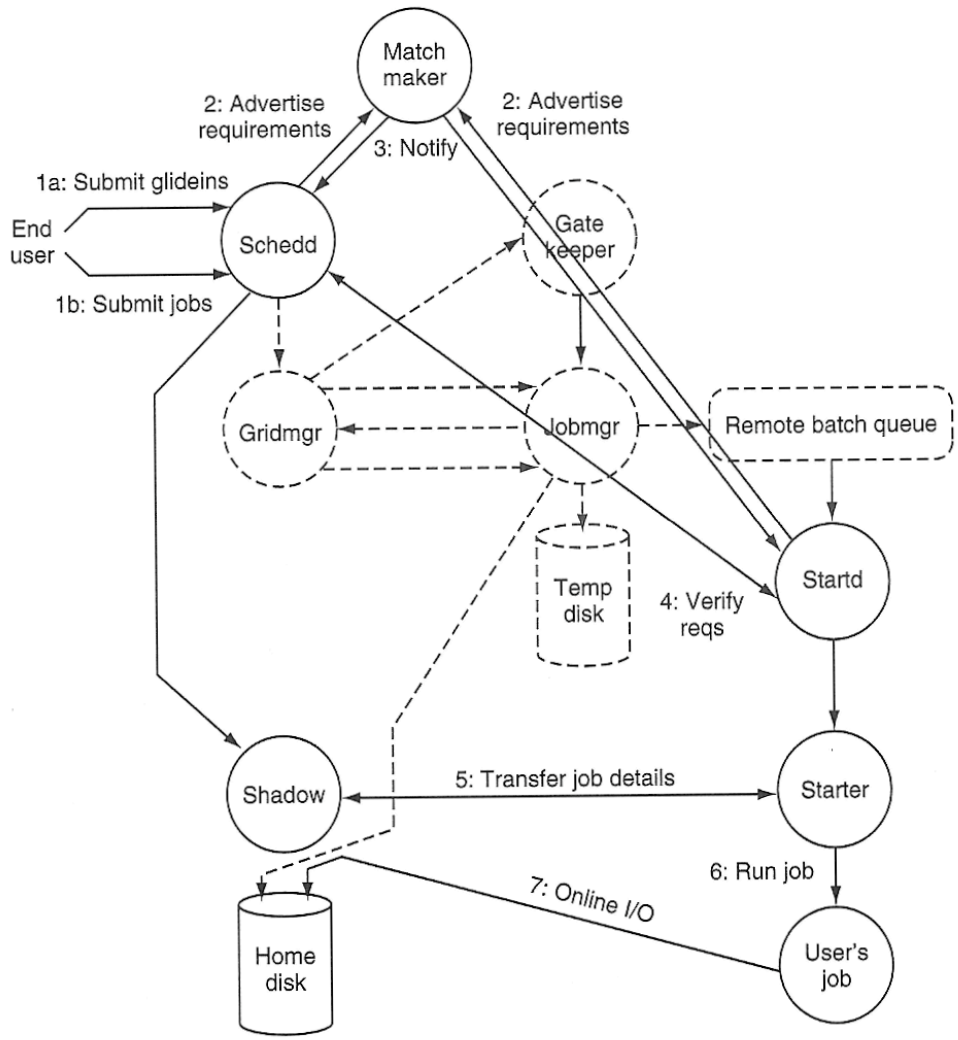
Condor-G: Condor for the Grid



7 steps to run a Condor-G job

- Condor-G can cross institutional boundaries
- Supports both Condor and non-Condor pools
- Uses
  - Grid Security Infrastructure (GSI)
  - Grid Resource Allocation Manager (GRAM)
  - Global Access to Storage (GASS)

Condor-G Glidein



- Condor pool on top of a Condor-G system
- User submits glideins via Condor-G over GRAM
  - A glidein is a Condor deamon
  - Each glidein creates (part of) an ad-hoc Condor pool
- User submits Condor jobs to run on the pool
- Matchmaker more complex
  - Can be user-defined

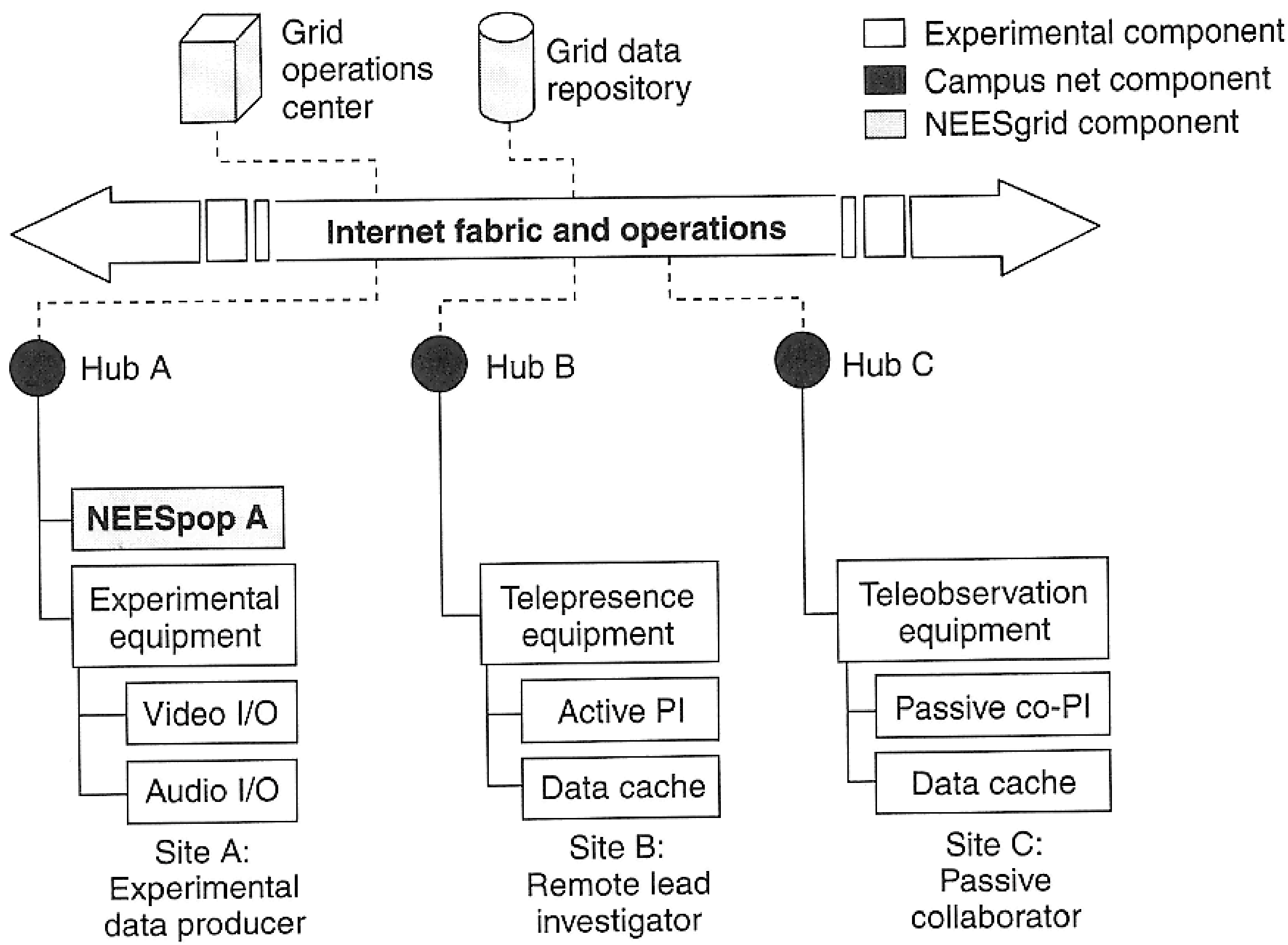
Two Example Applications

- NEESgrid Earthquake Engineering Collaboratory
- The Virtual Observatory World-Wide Telescope

NEESgrid

- Used by the earthquake engineering community
  - http://it.nees.org
- Goals is to enhance the utility of experimental facilities
  - Sharing of experimental data
  - Collaboration between experts from different domains
- Five basic functional components
  - Collaboration technology
  - Information management for sharing data
  - Simulation systems and software
  - NEESgrid-enabled desktops for direct access to Grid
  - Support nodes which maintain online knowledge bases

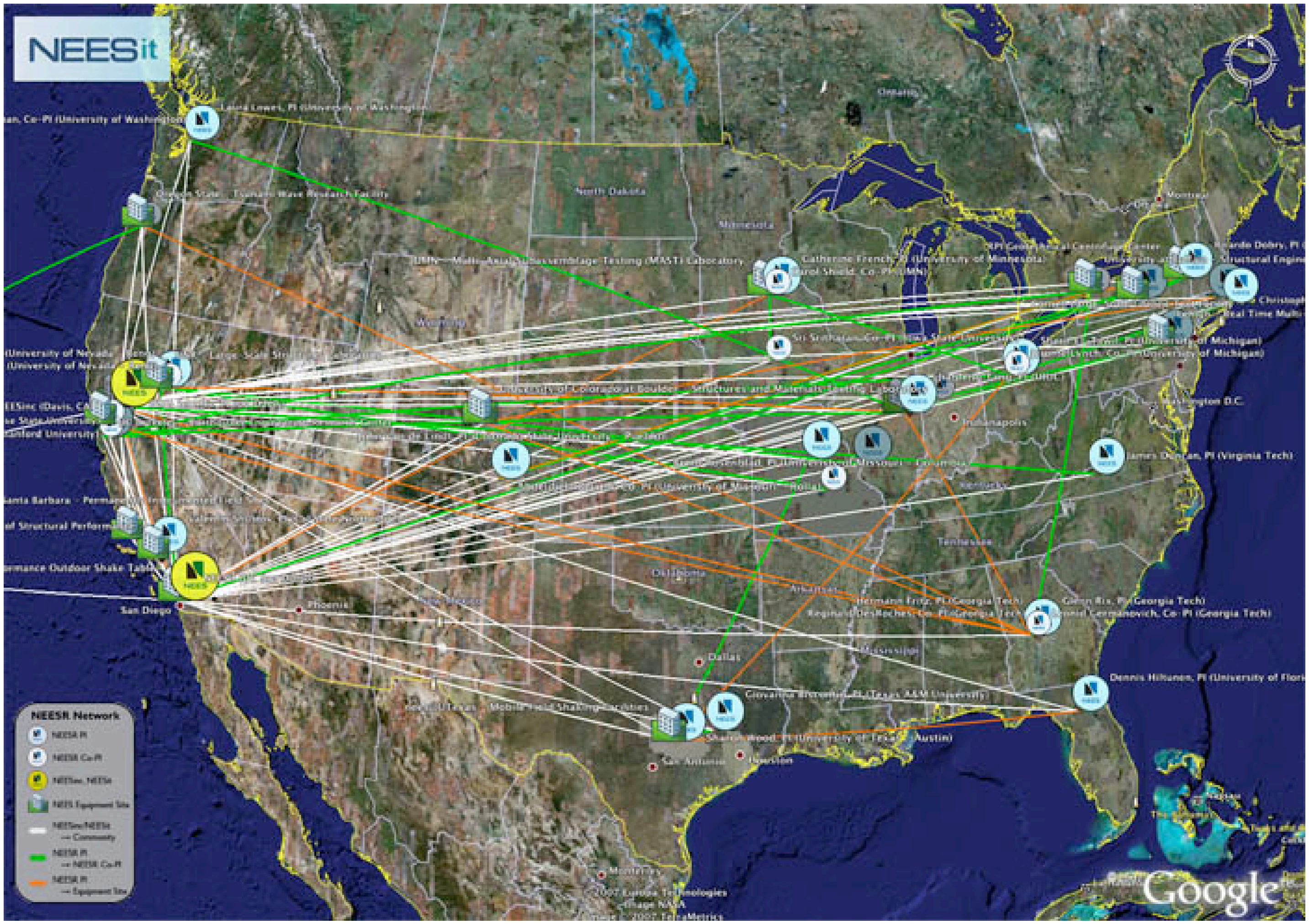
NEESGrid Overview



NEESPoP

- NEESPoP (NEES Point of Presence) deployed at each experimental facility
- Provides
  - Site status published via Globus MDS
  - Teleobservation using streaming data services protocol (NSDS)
  - Telecontrol of actuators using transaction-oriented NEES telecontrol protocol (NTCP)
  - Collaboration services (via so-called CHEF portlets)
    - Forums, electronic notebooks
    - Metadata browser for editing and publishing of metadata associated with experimental and simulation data
    - Remote administration services
    - Data access via GridFTP

NEESgrid and Google Earth



Earthquake engineering-related points of interest

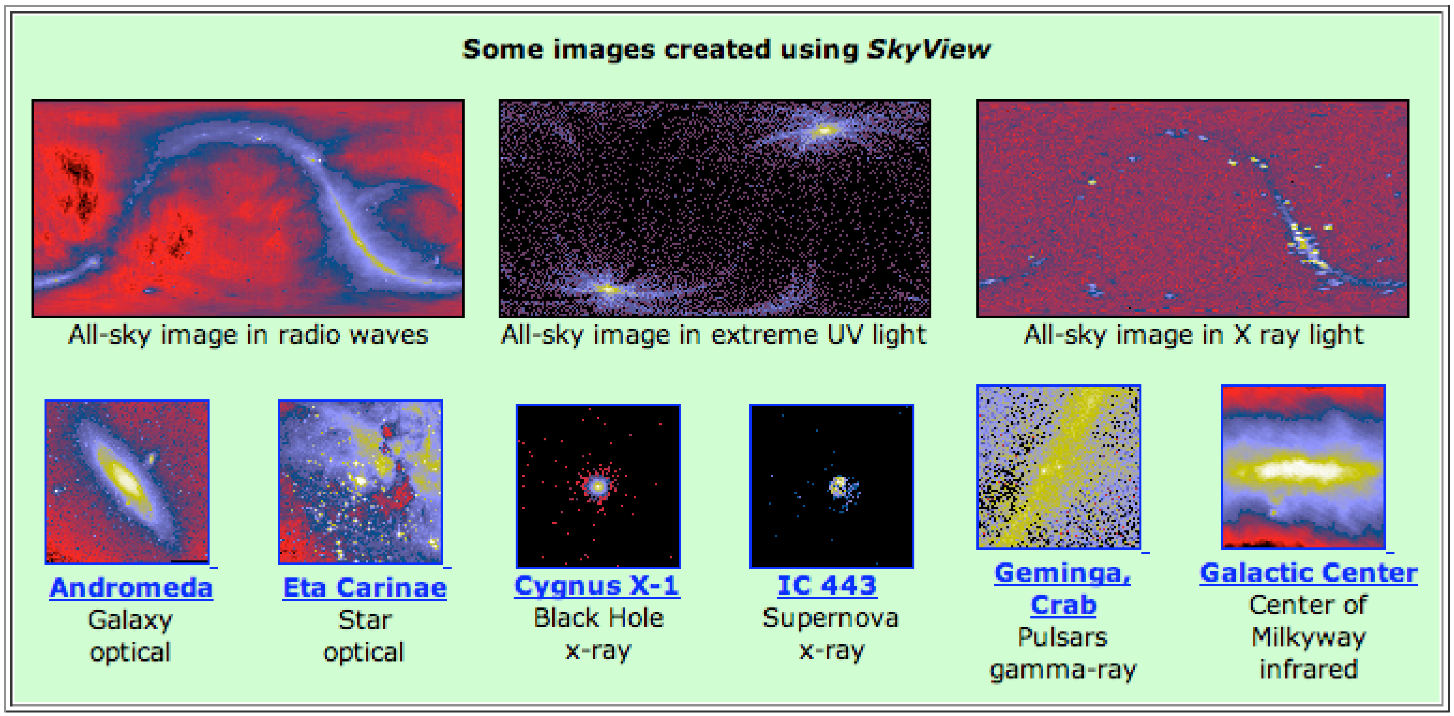
The Virtual Observatory World-Wide Telescope

- Unify astronomy archives into giant database
- Integrate archives as a single “intelligent” telescope
- Hope for new discoveries through federating datasets from multiple independent projects

- Hierarchical architecture
  - Archives: raw data (images/text) organized with metadata
  - XML Web services: on-demand queries and a XML schema data model (VOTable) to represent astronomy data
  - Registries and portals: record what kind of data the archive provides (sky coverage, temporal coverage, spectral coverage, and resolution)

Virtual Observatory Tools

- Digital sky surveys are created using the Chimera virtual data toolkit (built on Globus and Condor)
- Virtual Sky provides image navigation over the whole sky, e.g. access via http://skyview.gsfc.nasa.gov



Further Reading

- [Grid] Chapter 4
- Optional:
  - NEESgrid: [Grid] Chapter 6
  - Virtual Observatory: [Grid] Chapter 7
  - Condor and reliability: [Grid] Chapter 19