HPC & BigData
Grid Computing

High Performance computing Curriculum

UvA-SARA

http://www.hpc.uva.nl/

outline

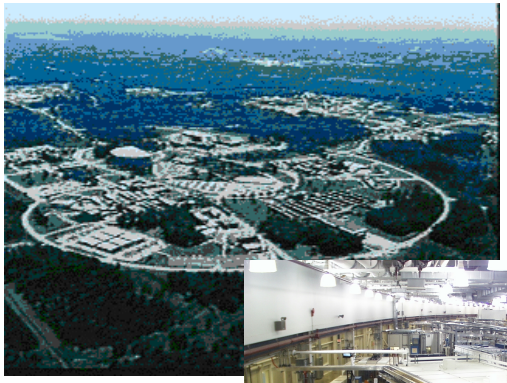
- e-Science
- Grid approach
- Grid computing
- Programming models for the Grid
- Grid-middleware
- Web Services
- Open Grid Service Architecture (OGSA)

Doing Science in the 21th century

- Nowadays Scientific Applications are
  - CPU intensive
  - Produce/process Huge sets of Data
  - Requires access to *geographically distributed* and *expensive* instruments

Online Access to Scientific Instruments

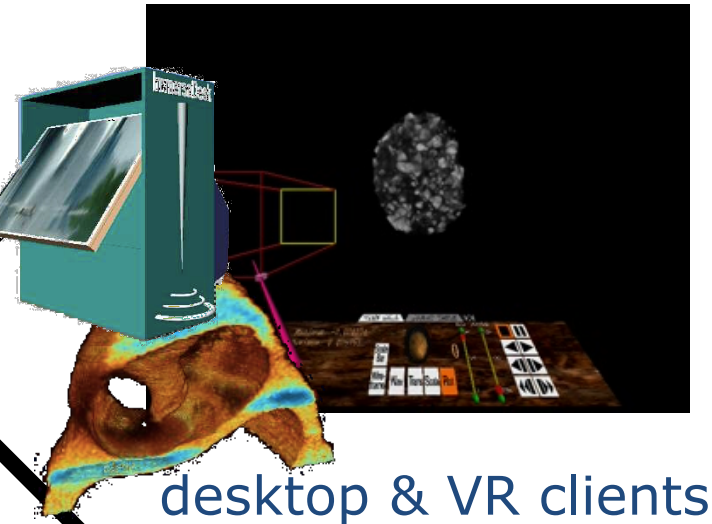
Advanced Photon Source



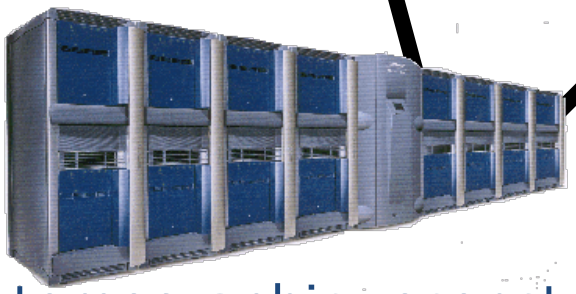
real-time collection

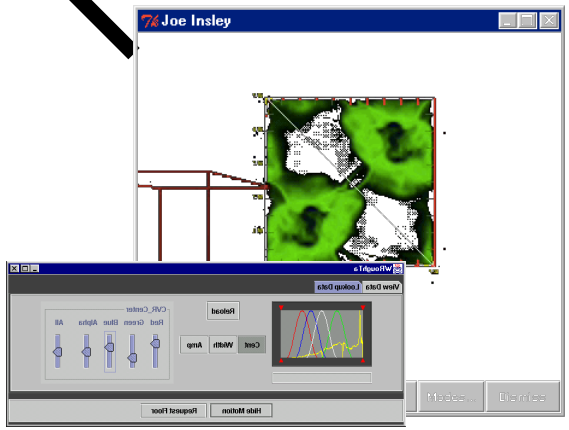
wide-area dissemination

archival storage



desktop & VR clients with shared controls





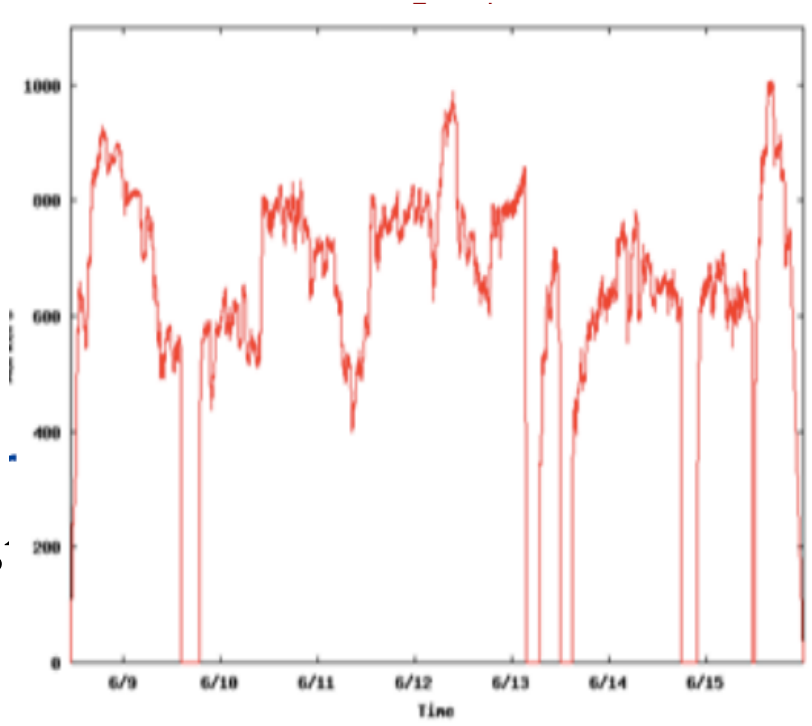
tomographic reconstruction

DOE X-ray grand challenge: ANL, USC/ISI, NIST, U.Chicago

From the Grid tutorials available at : http:.//www.globus.org

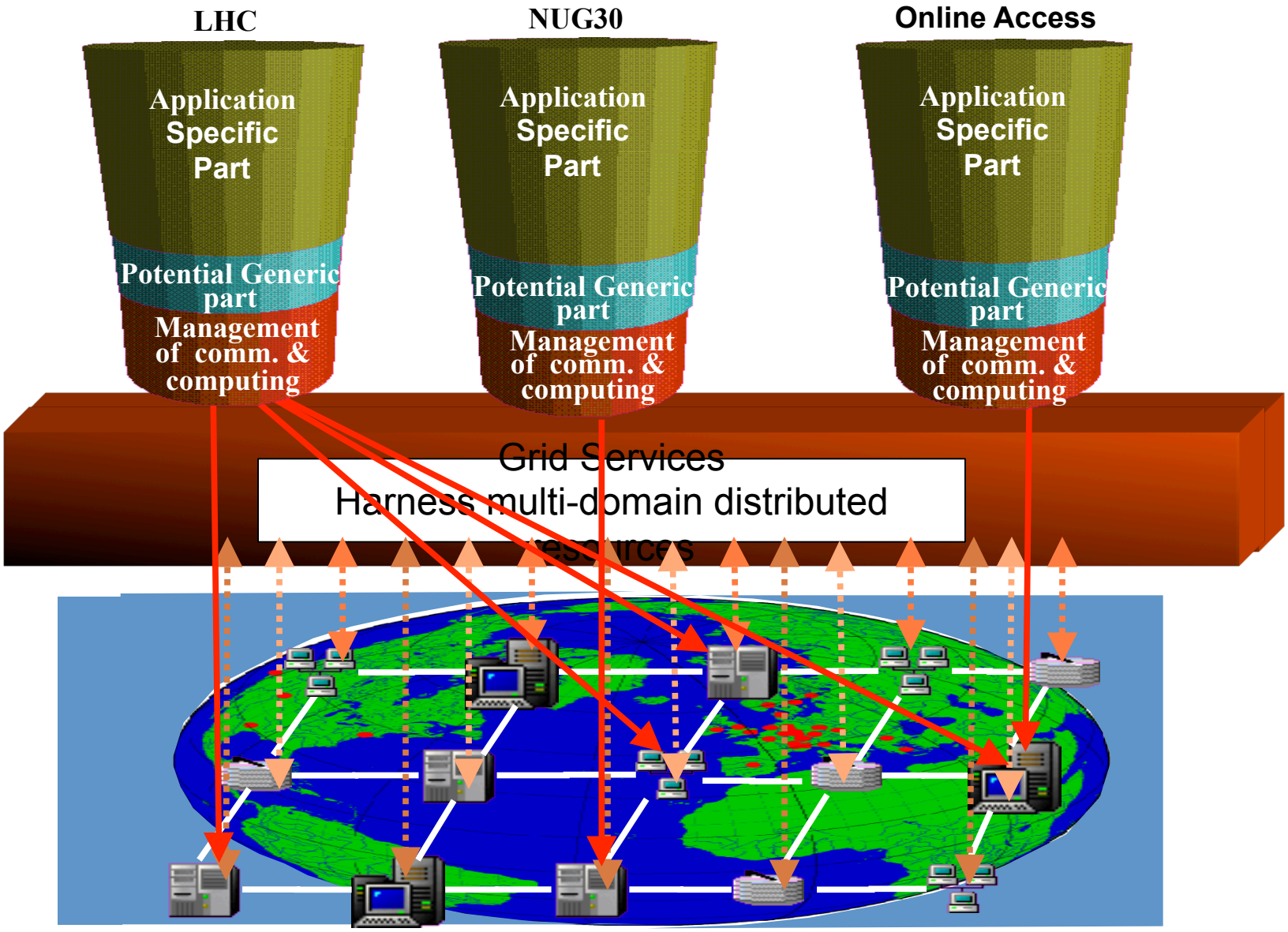
CPU intensive Science: Optimization problem NUG30

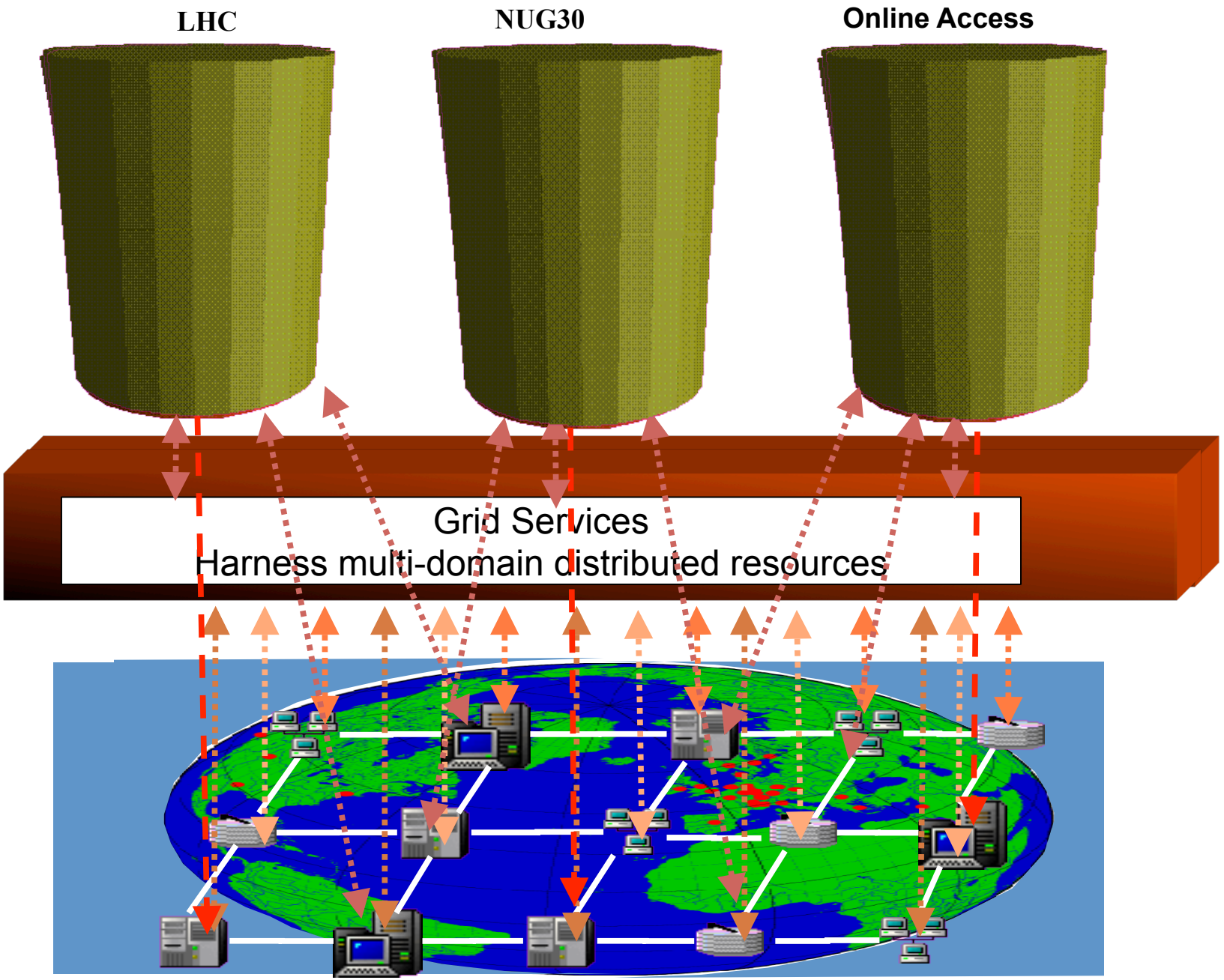
- The problem, a quadratic assignment problem (QAP) known as NUG30
  - given a set of n locations and n facilities, the goal is to assign each facility to a location.
  - There are **n!** possible assignments
- NUG30 proposed in 1968 as a tes of computer capabilities, but remained unsolved because of its great complexity.



Nug30 Quadratic Assignment Problem Solved by 1,000 https://scout.wisc.edu/archives/r7125

To solve these problems?




LHC
NUG30
Online Access
Grid Services
Harness multi-domain distributed resources

outline

- e-Science
- Grid approach
- Grid computing
- Programming models for the Grid
- Grid-middleware
- Web Services
- Open Grid Service Architecture (OGSA)

The Grid Problem

- Flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources

- Enable communities ("Virtual Organizations") to share geographically distributed resources as they pursue common goals -- *assuming the absence* of : central location, central control, existing trust relationships.

From the Grid tutorials available at : http:.//www.globus.org

Some Definitions of the Grid?

"A Computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities". Karl Kesselman & Ian Foster.

"The overall motivation for Grids is to enable the routine interactions of resources geographically and organizationally dispersed to facilitate Large-scale Science and engineering" The Vision for a DOE Science Grid, William Johnston, Lawrence Berkeley Nat. Lab.

"Making possible a shared large wide-area Computational infrastructure a concept which has been named the Grid" Peter Dinda, Gorgia Tech, 2001.

The real Grid target

- A Grid is a system that is able to
  - Coordinate resources
    - not subject to centralized control
  - Use standard, open, general-purpose protocols and interfaces
  - Deliver nontrivial qualities of service.

"Ian Foster's 3 point checklist"

Coordinated Sharing

- The sharing is controlled by the providers and consumers
  - what is shared?
  - who is allowed to share?
  - and the conditions under which sharing occurs?

- sharing relationships
  - client-server, peer-to-peer, and brokered
  - access control: fine AC, delegation, local/global policies

From "The Anatomy of the Grid: Enabling Scalable Virtual Organizations" Foster et al

outline

- e-Science
- Grid approach
- Grid computing
- Programming models for the Grid
- Grid-middleware
- Web Services
- Open Grid Service Architecture (OGSA)

What is Grid Computing

- Grid computing is the use of hundreds, thousands, or millions of geographically and organizationally disperse and diverse resources to solve:

➔ problems that require more computing power than is available from a single machine or from a local area distributed system

Potential Grid Application

- An application which requires the grid solution is likely distributed (Distributed Computing) and fit in one of the following paradigms:
  - High throughput Computing
  - High performance Computing

Grid computing will be mainly needed for large-scale, high-performance computing.

Distributed Computing

- Distributed computing is a programming model in which processing occurs in many geographically distributed places.
  - Processing can occur wherever it makes the most sense, whether that is on a server, Web site, personal computer, etc.

- Distributed computing and grid computing either
  - overlap or distributed computing is a subset of grid computing

From "The Anatomy of the Grid: Enabling Scalable Virtual Organizations" Foster et al

High Throughput Computing

- HTC employs large amounts of computing power for very lengthy periods
  - HTC is needed for doing sensitivity analyses, parametric studies or simulations to establish statistical confidence.
- The features of HTC are
  - Availability of computing power for a long period of time
  - Efficient fault tolerance mechanism
- The key to HTC in grids
  - Efficiently harness the use of all available resources across organizations

High Performance Computing

- HPC brings enormous amounts of computing power to bear over relatively short periods of time.
  - HPC is needed for decision-support or applications under sharp time-constraint, such as weather modeling
- HPC applications are:
  - Large in scale and complex in structure.
  - Real time requirements.
  - Ultimately must run on more than one type of HPC system.

HPC/HTC requirements

- HPC/HTC requires a balance of computation and communication among all resources involved.
  - Managing computation,
  - communication,
  - data locality

outline

- e-Science
- Grid approach
- Grid computing
- Programming models for the Grid
- Grid-middleware
- Web Services
- Open Grid Service Architecture (OGSA)

Programming Model for the grid

- To achieve petaflop rates on tightly/loosely coupled grid clusters, applications will have to allow:
  - extremely large granularity or produce massive parallelism such that high latencies can be tolerated.
- This type of parallelism, and the performance delivered by it in a heterogeneous environment, is
  - currently manageable by hand-coded applications

Programming Model for the grid

- A programming model can be presented in different forms: a language, a library API, or a tool with extensible functionality.

- The successful programming model will
  - enable both high-performance and the flexible composition and management of resources.
  - influence the entire software lifecycle: design, implementation, debugging, operation, maintenance, etc.
  - facilitate the effective use of all manner of development tools, e.g., compilers, debuggers, performance monitors, etc

Grid Programming Issues

- Portability, Interoperability, and Adaptability
- Discovery
- Performance
- Fault Tolerance
- Security

Programming models

- Shared-state models
- Message passing models
- RPC and RMI models
- Hybrid Models
- Peer to Peer Models
- Web Service Models
- …

outline

- e-Science
- Grid approach
- Grid computing
- Programming models for the Grid
- Grid-middleware
- Web Services
- Open Grid Service Architecture (OGSA)

Grid Middleware Definition

- Architecture identifies the fundamental system components, specifies purpose and function of these components, and indicates how these components interact with each other.
- Grid architecture is a protocol architecture, with protocols defining the basic mechanisms by which VO users and resources negotiate, establish, manage and exploit sharing relationships.
- Grid architecture is also a service standard-based open architecture that facilitates extensibility, interoperability, portability and code sharing.

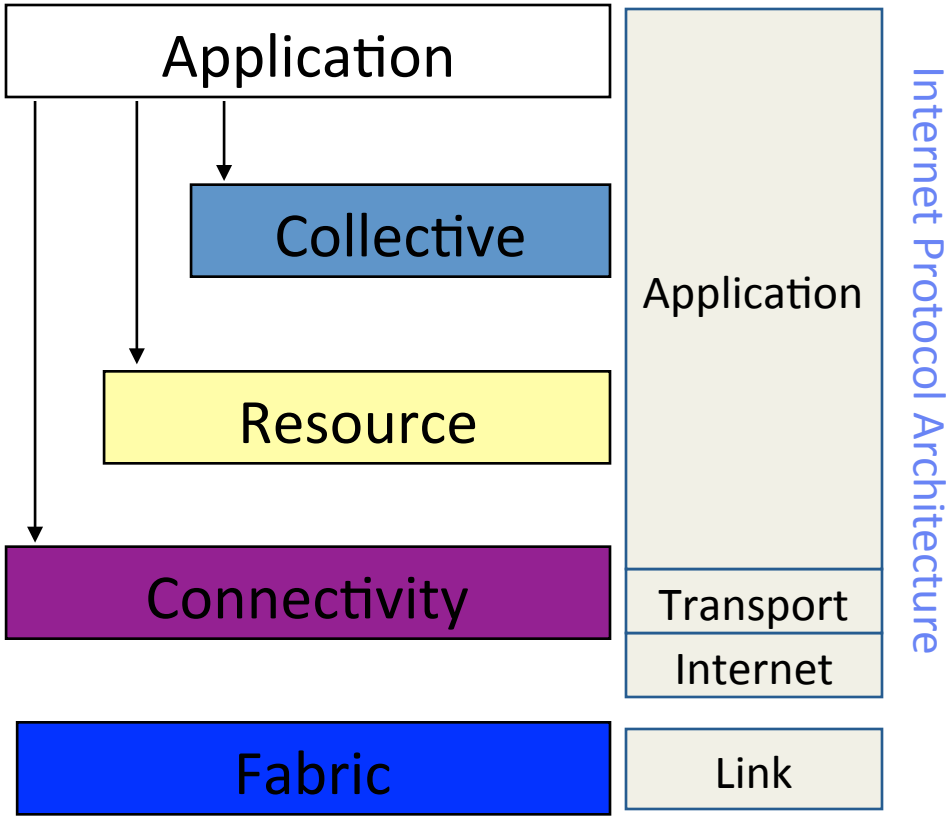
Architecture

"Coordinating multiple resources": ubiquitous infrastructure services, app-specific distributed services

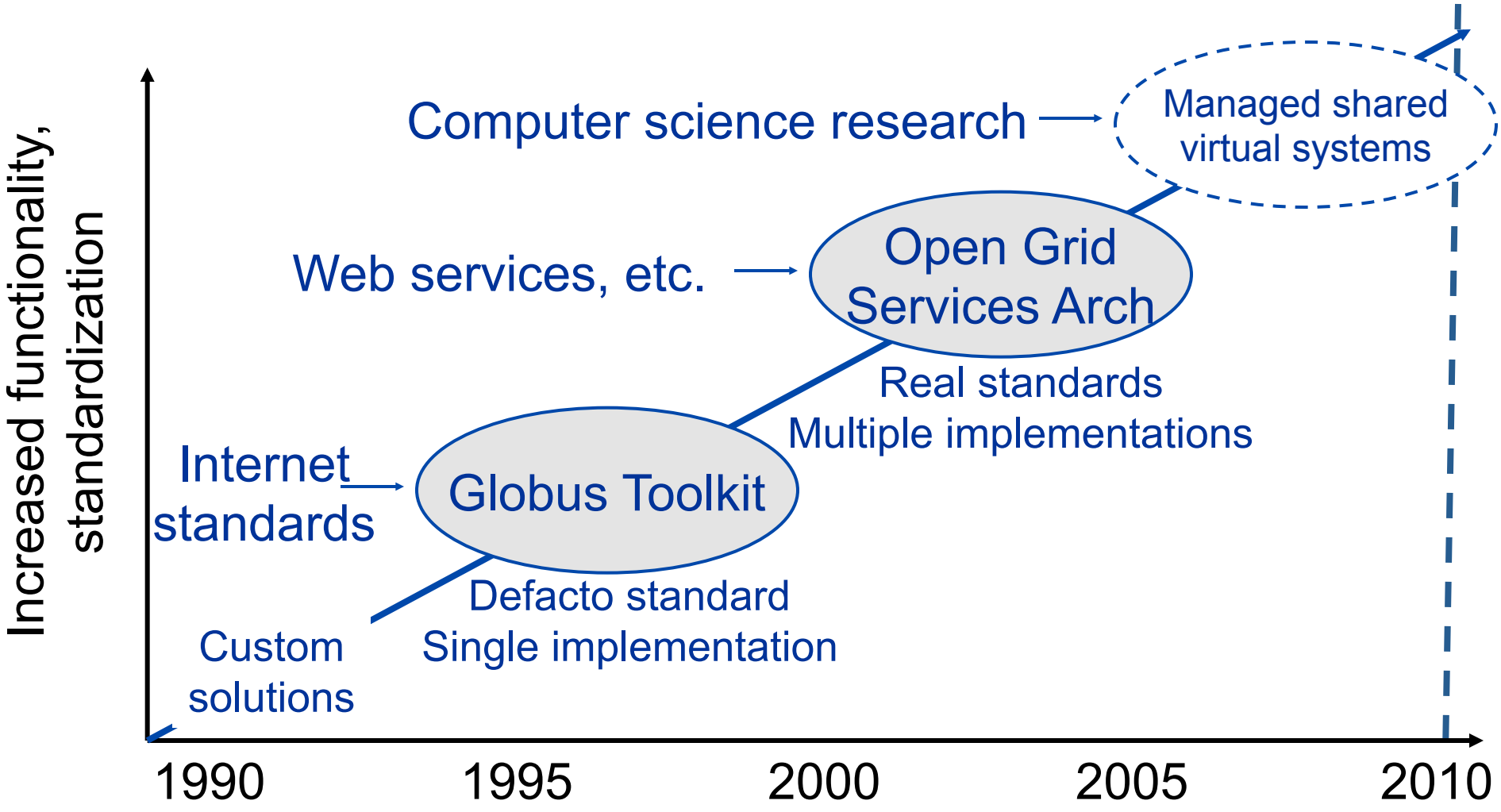
"Sharing single resources": negotiating access, controlling use

"Talking to things": communication (Internet protocols) & security

"Controlling things locally": Access to, & control of resources



Emergence of Open Grid Standards



"Grid Computing and Scaling Up the Internet" I. Foster, IPv6 Forum, an

Examples of Grid Middleware

- Globus Toolkit (GT4.X) now (GT5.X)
  - www.globus.org
- Legion/Avaki
  - http://www.avaki.com/
  - http://legion.virginia.edu/
- Grid Sun engine
  - http://www.sun.com/service/sungrid/overview.jsp
- Unicore
  - http://www.unicore.org

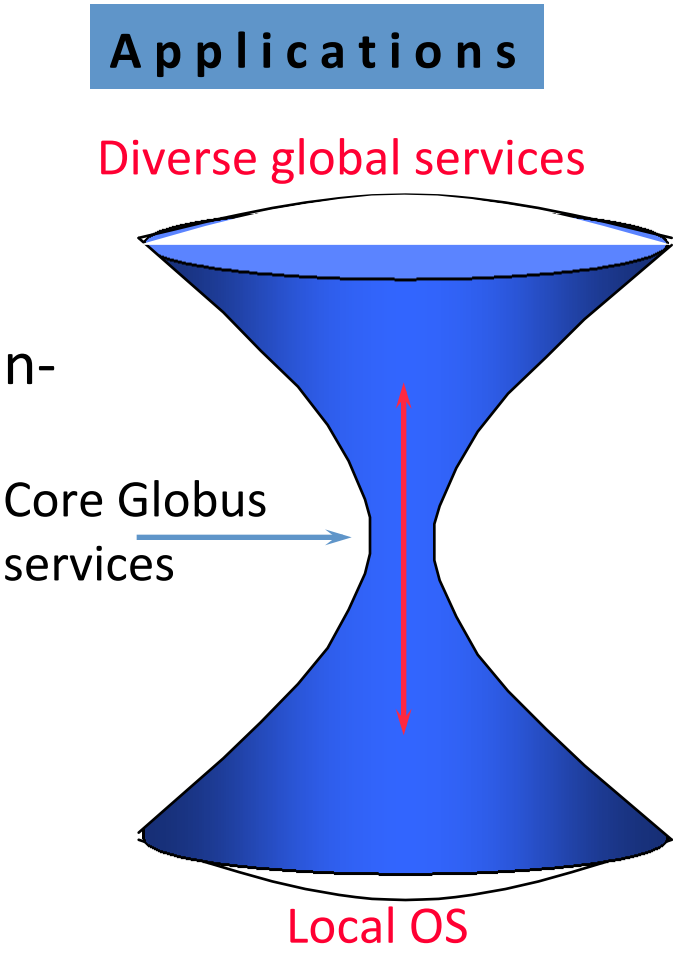
The Grid Middleware

- Software toolkit addressing key technical areas
  - Offer a modular "bag of technologies"
  - Enable incremental development of grid-enabled tools and applications
  - Define and standardize grid protocols and APIs
- Focus is on inter-domain issues, not clustering
  - Collaborative resource use spanning multiple organizations
  - Integrates cleanly with intra-domain services
  - Creates a "collective" service layer

"Basics Globus Toolkit™ Developer Tutorial" Globus Team, 2003

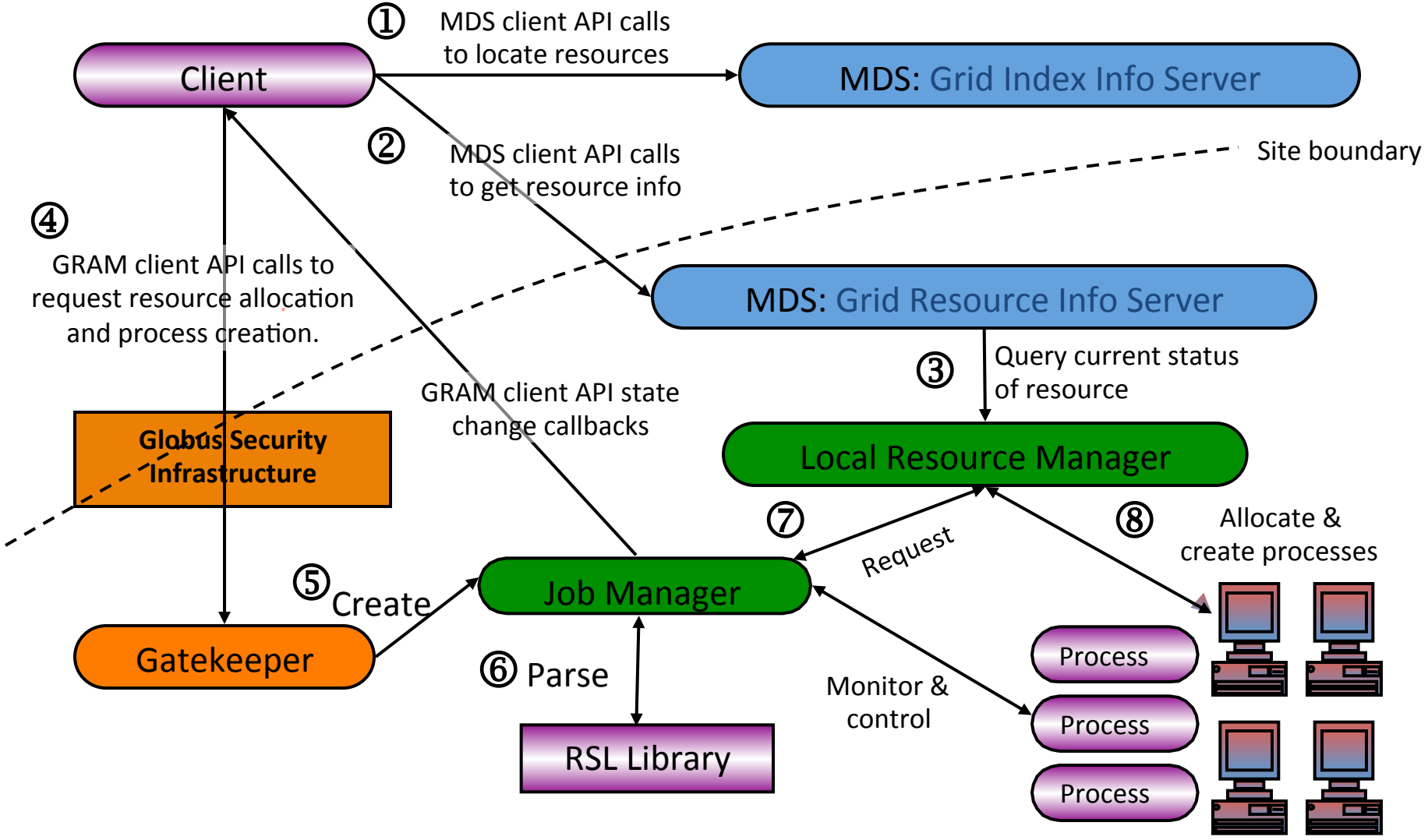
Globus Approach

- Focus on architecture issues
  - Provide implementations of grid protocols and APIs as basic infrastructure
  - Use to construct high-level, domain-specific solutions
- Design principles
  - Keep participation cost low
  - Enable local control
  - Support for adaptation



"Basics Globus Toolkit™ Developer Tutorial" Globus Team, 2003

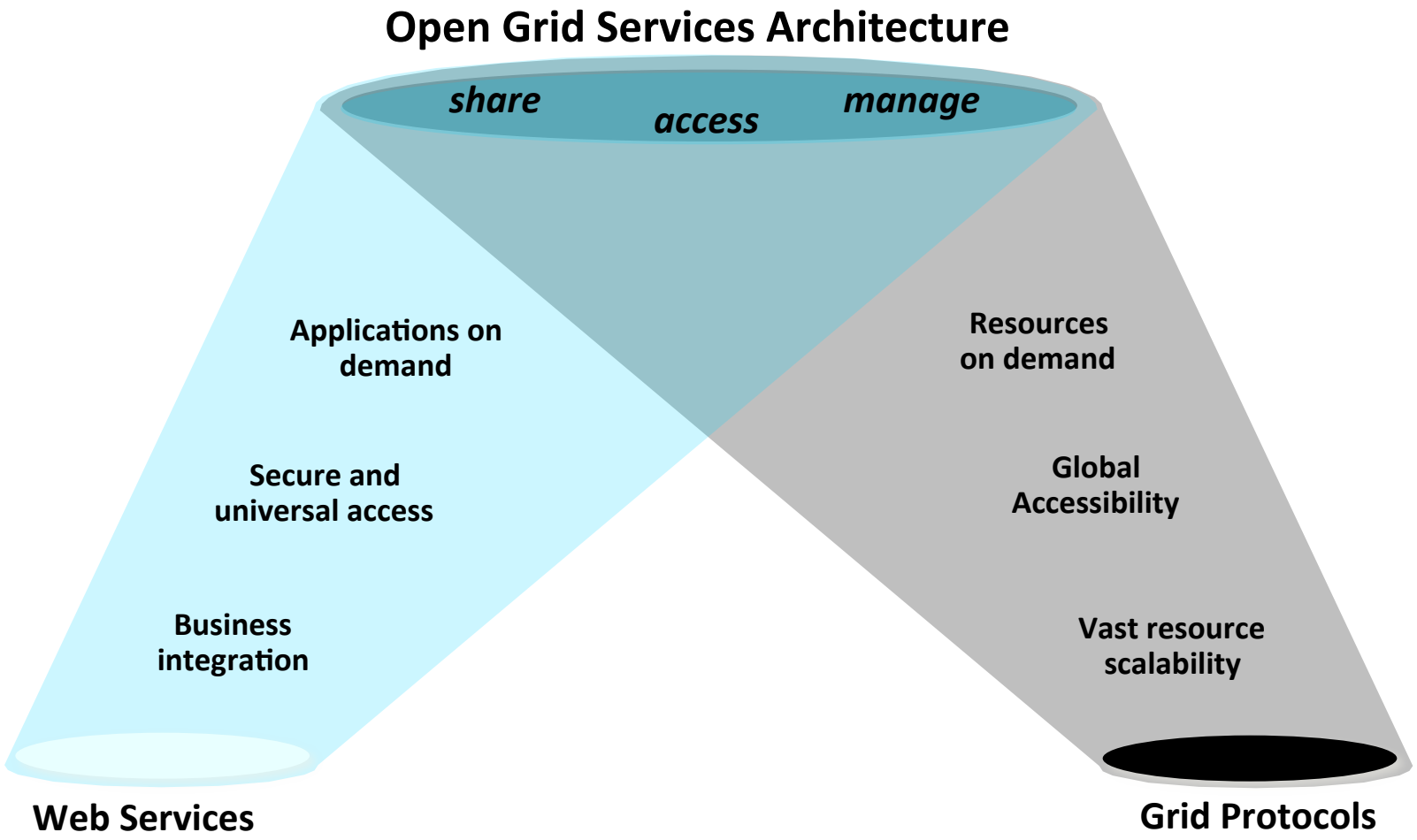
Globus Toolkit 2.0 Components



outline

- e-Science
- Grid approach
- Grid computing
- Programming models for the Grid
- Grid-middleware
- Web Services
- Open Grid Service Architecture (OGSA)

Best of Two Worlds



'Open Grid Services Architecture Evolution, J.P. Prost, IBM Montpellier, France, Ecole Bruide 2004

Web Services

- Increasingly popular standards-based framework for accessing network applications
  - W3C standardization; Microsoft, IBM, Sun, others
- WSDL: Web Services Description Language
  - Interface Definition Language for Web services
- SOAP: Simple Object Access Protocol
  - XML-based RPC protocol; common WSDL target
- WS-Inspection
  - Conventions for locating service descriptions
- UDDI: Universal Desc., Discovery, & Integration
  - Directory for Web services

"Globus Toolkit Futures: An Open Grid Services Architecture" Ian Foster et al. Globus Tutorial, Argonne National Laboratory, January 29, 2002

The Need to Support Transient Service Instances

- "Web services" address discovery & invocation of persistent services
  - Interface to persistent state of entire enterprise
- In Grids, must also support transient service instances, created/destroyed dynamically
  - Interfaces to the states of distributed activities
  - E.g. workflow, video conf., dist. data analysis
- Significant implications for how services are managed, named, discovered, and used
  - In fact, much of the work is concerned with the management of service instances

"Globus Toolkit Futures: An Open Grid Services Architecture" Ian Foster et al. Globus Tutorial, Argonne National Laboratory, January 29, 2002

outline

- e-Science
- Grid approach
- Grid computing
- Programming models for the Grid
- Grid-middleware
- Web Services
- Open Grid Service Architecture (OGSA)

Open Grid Services Architecture

- Service orientation to virtualize resources
- From Web services:
  - Standard interface definition mechanisms: multiple protocol bindings, multiple implementations, local/remote transparency
- Building on Globus Toolkit:
  - Grid service: semantics for service interactions
  - Management of transient instances (& state)
  - Factory, Registry, Discovery, other services
  - Reliable and secure transport
- Multiple hosting targets: J2EE, .NET, …

"Globus Toolkit Futures: An Open Grid Services Architecture" Ian Foster et al. Globus Tutorial, Argonne National Laboratory, January 29, 2002

Open Grid Services Architecture Objectives

- Manage resources across distributed heterogeneous platforms
- Deliver seamless QoS
- Provide a common base for autonomic management solutions
- Define open, published interfaces
- Exploit industry-standard integration technologies
  - Web Services, SOAP, XML,...
- Integrate with existing IT resources

'Open Grid Services Architecture Evolution, J.P. Prost, IBM Montpellier, France, Ecole Bruide 2004