# Introduction to Globus and OGSA

ESRIN Grid Tutorial

10th December 2003

# Globus Toolkit™

- A software toolkit addressing key technical problems in the development of Grid enabled tools, services, and applications
  - Offer a modular "bag of technologies"
  - Enable *incremental* development of grid-enabled tools and applications
  - Implement standard Grid protocols and APIs
  - Make available under liberal open source license

# General Approach

- Define Grid protocols & APIs
  - Protocol-mediated access to remote resources
  - Integrate and extend existing standards
  - "On the Grid" = speak "Intergrid" protocols
- Develop a reference implementation
  - Open source Globus Toolkit
  - Client and server SDKs, services, tools, etc.
- Grid-enable wide variety of tools
  - Globus Toolkit, FTP, SSH, Condor, SRB, MPI, …
- Learn through deployment and applications

# Key Protocols

- The Globus Toolkit™ centers around four key protocols
  - Connectivity layer:
    - > *Security*: Grid Security Infrastructure (GSI)
  - Resource layer:
    - > *Resource Management*: Grid Resource Allocation Management (GRAM)
    - > *Information Services*: Grid Resource Information Protocol (GRIP)
    - > *Data Transfer*: Grid File Transfer Protocol (GridFTP)
- Also key collective layer protocols
  - Info Services, Replica Management, etc.

# The Globus Toolkit™:
# Security

# Security Terminology

- Authentication: Establishing identity
- Authorization: Establishing rights
- Message protection
  - Message integrity
  - Message confidentiality
- Non-repudiation
- Digital signature
- Accounting
- Certificate Authority (CA)

# Grid Security Infrastructure (GSI)

- Extensions to standard protocols & APIs
  - Standards: SSL/TLS, X.509 & CA, GSS-API
  - Extensions for single sign-on and delegation
- Globus Toolkit reference implementation of GSI
  - SSLeay/OpenSSL + GSS-API + SSO/delegation
  - Tools and services to interface to local security
    - > Simple ACLs; SSLK5/PKINIT for access to K5, AFS; …
  - Tools for credential management
    - > Login, logout, etc.
    - > Smartcards
    - > MyProxy: Web portal login and delegation
    - > K5cert: Automatic X.509 certificate creation

# Review of Public Key Cryptography

- Asymmetric keys
  - A **private** key is used to encrypt data.
  - A **public** key can decrypt data encrypted with the private key.
- An X.509 certificate includes…
  - Someone's subject name (user ID)
  - Their public key
  - A "signature" from a Certificate Authority (CA) that:
    - > Proves that the certificate came from the CA.
    - > Vouches for the subject name
    - > Vouches for the binding of the public key to the subject

# Public Key Based Authentication

- User sends certificate over the wire.
- Other end sends user a challenge string.
- User encodes the challenge string with private key
  - Possession of private key means you can authenticate as subject in certificate
- Public key is used to decode the challenge.
  - If you can decode it, you know the subject
- Treat your private key carefully!!
  - Private key is stored only in well-guarded places, and only in encrypted form

# X.509 Proxy Certificate

- Defines how a short term, restricted credential can be created from a normal, long-term X.509 credential
  - A "proxy certificate" is a special type of X.509 certificate that is signed by the normal end entity cert, or by another proxy
  - Supports single sign-on & delegation through "impersonation"
  - Currently an IETF draft

# User Proxies

- Minimize exposure of user's private key
- A temporary, X.509 proxy credential for use by our computations
  - We call this a user proxy certificate
  - Allows process to act on behalf of user
  - User-signed user proxy cert stored in local file
  - Created via "grid-proxy-init" command
- Proxy's private key is not encrypted
  - Rely on file system security, proxy certificate file must be readable only by the owner

# Delegation

- Remote creation of a user proxy
- Results in a new private key and X.509 proxy certificate, signed by the original key
- Allows remote process to act on behalf of the user
- Avoids sending passwords or private keys across the network

# Security Summary

- GSI successfully addresses wide variety of Grid security issues
- Broad acceptance, deployment, integration with tools
- Standardization on-going in IETF & GGF
- Ongoing R&D to address next set of issues
- For more information:
  - www.globus.org/research/papers.html
    - "A Security Architecture for Computational Grids"
    - "Design and Deployment of a National-Scale Authentication Infrastructure"
  - www.gridforum.org/security

# The Globus Toolkit™:
# Resource Management

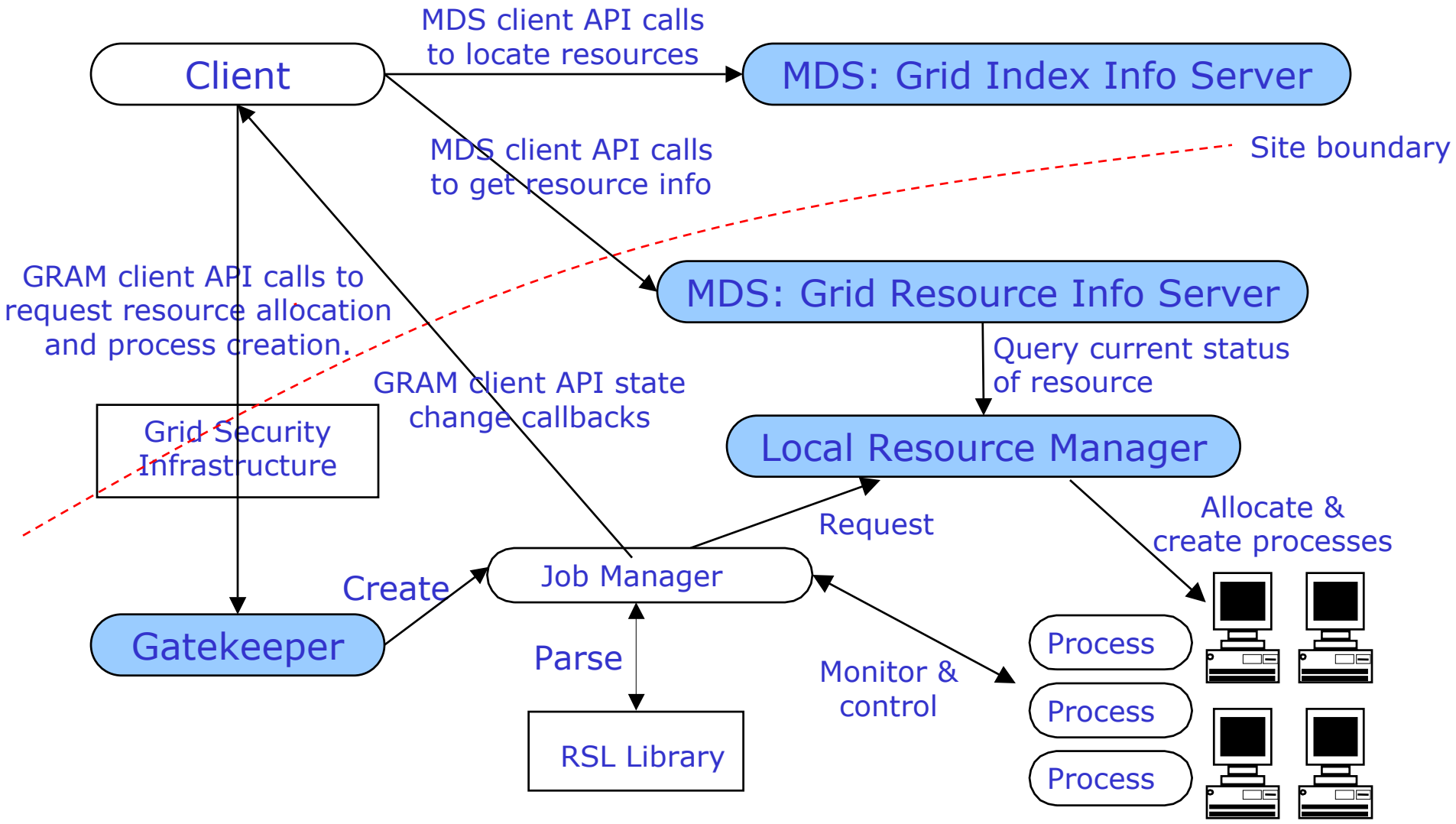
# Resource Management

- The Grid Resource Allocation Management (GRAM) protocol and client API allows programs to be started on remote resources, despite local heterogeneity
- Resource Specification Language (RSL) is used to communicate requirements
- A layered architecture allows application-specific resource brokers and co-allocators to be defined in terms of GRAM services
  - Integrated with Condor, PBS, MPICH-G2, …

# Globus Toolkit Implementation

- Gatekeeper
  - Single point of entry
  - Authenticates user, maps to local security environment, runs service
  - In essence, a “secure inetd”
- Job manager
  - A gatekeeper service
  - Layers on top of local resource management system (e.g., PBS, LSF, etc.)
  - Handles remote interaction with the job

# GRAM Components



# Job Submission Interfaces

- Globus Toolkit includes several command line programs for job submission
  - globus-job-run: Interactive jobs
  - globus-job-submit: Batch/offline jobs
  - globusrun: Flexible scripting infrastructure
- Others are building better interfaces
  - General purpose
    - Condor-G, PBS, GRD, Hotpage, etc
  - Application specific
    - ECCE', Cactus, Web portals

# globus-job-run

- For running of interactive jobs
- Additional functionality beyond rsh
  - Ex: Run 2 process job w/ executable staging

    ```
    globus-job-run -: host –np 2 –s myprog arg1 arg2
    ```
  - Ex: Run 5 processes across 2 hosts

    ```
    globus-job-run \
            -: host1 –np 2 –s myprog.linux arg1 \
            -: host2 –np 3 –s myprog.aix arg2
    ```
  - For list of arguments run:

    ```
    globus-job-run -help
    ```

# globus-job-submit

- For running of batch/offline jobs
  - globus-job-submit — Submit job
    - > Same interface as globus-job-run
    - > Returns immediately
  - globus-job-status — Check job status
  - globus-job-cancel — Cancel job
  - globus-job-get-output — Get job stdout/err
  - globus-job-clean — Cleanup after job

# globusrun

- Flexible job submission for scripting
  - Uses an RSL string to specify job request
  - Contains an embedded globus-gass-server
    - > Defines GASS URL prefix in RSL substitution variable: (stdout=$(GLOBUSRUN_GASS_URL)/stdout)
  - Supports both interactive and offline jobs
- Complex to use
  - Must write RSL by hand
  - Must understand its esoteric features
  - Generally you should use globus-job-* commands instead

# The Globus Toolkit™:
# Data Management

# Data Grid Problem

- "Enable a geographically distributed community [of thousands] to pool their resources in order to perform sophisticated, computationally intensive analyses on Petabytes of data"
- Note that this problem:
  - Is common to many areas of science
  - Overlaps strongly with other Grid problems

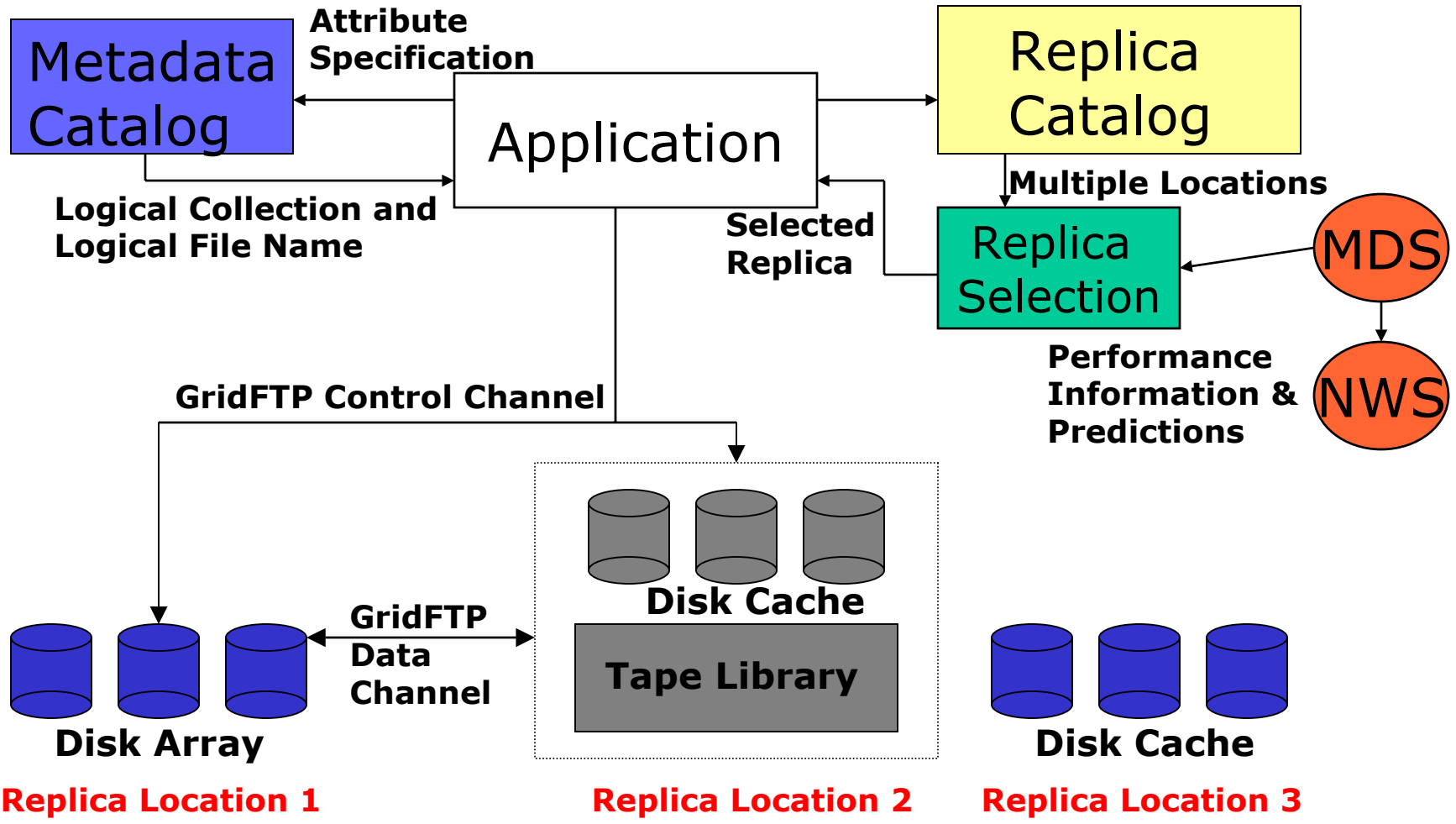
# Data Intensive Computing and Grids

- The term "Data Grid" is often used
  - Unfortunate as it implies a distinct infrastructure, which it isn't; but easy to say
- Data-intensive computing shares numerous requirements with collaboration, instrumentation, computation, …
  - Security, resource mgt, info services, etc.
- Important to exploit commonalities as very unlikely that multiple infrastructures can be maintained
- Fortunately this seems easy to do!

# Examples of Desired Data Grid Functionality

- High-speed, reliable access to remote data
- Automated discovery of "best" copy of data
- Manage replication to improve performance
- Co-schedule compute, storage, network
- "Transparency" wrt delivered performance
- Enforce access control on data
- Allow representation of "global" resource allocation policies

# A Model Architecture for Data Grids



# Globus Toolkit Components

Two major Data Grid components:

1. Data Transport and Access
   - Common protocol
     - Secure, efficient, flexible, extensible data movement
   - Family of tools supporting this protocol

2. Replica Management Architecture
   - Simple scheme for managing:
     - multiple copies of files
     - collections of files

# A Common, Secure, Efficient Data Access Protocol

- Common, *extensible* transfer protocol
  - Common protocol means all can interoperate
- Decouple low-level data transfer mechanisms from the storage service
- Advantages:
  - New, specialized storage systems are automatically compatible with existing systems
  - Existing systems have richer data transfer functionality
- Interface to many storage systems
  - HPSS, DPSS, file systems
  - Plan for SRB integration

# Access/Transport Protocol Requirements

- Suite of communication libraries and related tools that support
  - GSI, Kerberos security
  - Third-party transfers
  - Parameter set/negotiate
  - Partial file access
  - Reliability/restart
  - Large file support
  - Data channel reuse
  - Integrated instrumentation
  - Loggin/audit trail
  - Parallel transfers
  - Striping (cf DPSS)
  - Policy-based access control
  - Server-side computation
  - Proxies (firewall, load bal)
- All based on a standard, widely deployed protocol

# And The Protocol Is … GridFTP

- Why FTP?
  - Ubiquity enables interoperation with many commodity tools
  - Already supports many desired features, easily extended to support others
  - Well understood and supported
- We use the term GridFTP to refer to
  - Transfer protocol which meets requirements
  - Family of tools which implement the protocol
- Note GridFTP > FTP
- Note that despite name, GridFTP is not restricted to file transfer!

# The Globus Toolkit™:
# Information Services

# Grid Information Services

- System information is critical to operation of the grid and construction of applications
  - What resources are available?
    - > Resource discovery
  - What is the "state" of the grid?
    - > Resource selection
  - How to optimize resource use
    - > Application configuration and adaptation?
- We need a general information infrastructure to answer these questions

# Examples of Useful Information

- Characteristics of a compute resource
  - IP address, software available, system administrator, networks connected to, OS version, load
- Characteristics of a network
  - Bandwidth and latency, protocols, logical topology
- Characteristics of the Globus infrastructure
  - Hosts, resource managers

# Grid Information: Facts of Life

- Information is always old
  - Time of flight, changing system state
  - Need to provide quality metrics
- Distributed state hard to obtain
  - Complexity of global snapshot
- Component will fail
- Scalability and overhead
- Many different usage scenarios
  - Heterogeneous policy, different information organizations, different queries, etc.

# Grid Information Service

- Provide access to static and dynamic information regarding system components
- A basis for configuration and adaptation in heterogeneous, dynamic environments
- Requirements and characteristics
  - Uniform, flexible access to information
  - Scalable, efficient access to dynamic data
  - Access to multiple information sources
  - Decentralized maintenance

# Information Protocols

- Grid Resource Registration Protocol
  - Support information/resource discovery
  - Designed to support machine/network failure
- Grid Resource Inquiry Protocol
  - Query resource description server for information
  - Query aggregate server for information
  - LDAP V3.0 in Globus 1.1.3

# Metacomputing Directory Service

- Use LDAP as Inquiry
- Access information in a distributed directory
  - Directory represented by collection of LDAP servers
  - Each server optimized for particular function
- Directory can be updated by:
  - Information providers and tools
  - Applications (i.e., users)
  - Backend tools which generate info on demand
- Information dynamically available to tools and applications

# Two Classes Of MDS Servers

- Grid Resource Information Service (GRIS)
  - Supplies information about a specific resource
  - Configurable to support multiple information providers
  - LDAP as inquiry protocol
- Grid Index Information Service (GIIS)
  - Supplies collection of information which was gathered from multiple GRIS servers
  - Supports efficient queries against information which is spread across multiple GRIS server
  - LDAP as inquiry protocol

# MDS Components

- LDAP 3.0 Protocol Engine
  - Based on OpenLDAP with custom backend
  - Integrated caching
- Information providers
  - Delivers resource information to backend
- APIs for accessing & updating MDS contents
  - C, Java, PERL (LDAP API, JNDI)
- Various tools for manipulating MDS contents
  - Command line tools, Shell scripts & GUIs

# Searching an LDAP Directory

grid-info-search [options] filter [attributes]

- Default grid-info-search options
  - -h mds.globus.org — *MDS server*
  - -p 389 — *MDS port*
  - -b "o=Grid" — *search start point*
  - -T 30 — *LDAP query timeout*
  - -s **sub** — scope = subtree
    - *alternatives:*
      - **base** : *lookup this entry*
      - **one** : *lookup immediate children*

# Filtering

- Filters allow selection of object based on relational operators (=, ~=,<=, >=)
  - grid-info-search "cputype=*"
- Compound filters can be construct with Boolean operations: (&, |, !)
  - grid-info-search "(&(cputype=*)(cpuload1<=1.0))"
  - grid-info-search "(&(hn~=sdsc.edu)(latency<=10))"
- Hints:
  - white space is significant
  - use -L for LDIF format

# Example: Filtering

```
% grid-info-host-search -L "(objectclass=GlobusSoftware)"

dn: sw=Globus, hn=pitcairn.mcs.anl.gov, dc=mcs, dc=anl, dc=gov,
o=Grid
objectclass: GlobusSoftware
releasedate: 2000/04/11 19:48:29
releasemajor: 1
releaseminor: 1
releasepatch: 3
releasebeta: 11
lastupdate: Sun Apr 30 19:28:19 GMT 2000
objectname: sw=Globus, hn=pitcairn.mcs.anl.gov, dc=mcs, dc=anl,
dc=gov, o=Grid
```

# Example: Attribute Selection

% grid-info-host- search -L "(objectclass=*)" dn hn

- Returns the distinguished name (dn) and hostname (hn) of all objects

```
dn: sw=Globus, hn=pitcairn.mcs.anl.gov, dc=mcs, dc=anl, dc=gov, o=Grid

dn: hn=pitcairn.mcs.anl.gov, dc=mcs, dc=anl, dc=gov, o=Grid
hn: pitcairn.mcs.anl.gov

dn: service=jobmanager, hn=pitcairn.mcs.anl.gov, dc=mcs, dc=anl, dc=gov, o=Grid
hn: pitcairn.mcs.anl.gov

dn: queue=default, service=jobmanager, hn=pitcairn.mcs.anl.gov, dc=mcs, dc=anl,
    dc=gov, o=Grid
```

- Objects without hn fields are still listed
- DNs are always listed

# Example: Discovering CPU Load

- Retrieve CPU load fields of compute resources

% **grid-info-search -L "(objectclass=GlobusComputeResource)" **
**dn cpuload1 cpuload5 cpuload15**

dn: hn=lemon.mcs.anl.gov, ou=MCS, o=Argonne National Laboratory,
o=Globus, c=US
cpuload1: 0.48
cpuload5: 0.20
cpuload15: 0.03

dn: hn=tuva.mcs.anl.gov, ou=MCS, o=Argonne National Laboratory,
o=Globus, c=US
cpuload1: 3.11
cpuload5: 2.64
cpuload15: 2.57

# Open Grid Services Architecture

- Service-oriented architecture
  - Key to virtualization, discovery, composition, local-remote transparency
- Leverage industry standards
  - Internet, Web services
- Distributed service management
  - A "component model for Web services"
- A framework for the definition of composable, interoperable services

"The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration", Foster, Kesselman, Nick, Tuecke, 2002

# What is a Web service?

- A Web service is an entity that can be:
  - Described (using WSDL)
  - Published
  - Discovered
  - Invoked by a client
- W3C technology standardization process
- Often associated with specific technologies and implementations
  - Standards XML, WSDL,SOAP,UDDI
  - Implementations:WebSphere,.NET,others

# More Specifically

- A standard substrate: the Grid service
  - OGSI = Open Grid Service Infrastructure
  - Web services interfaces and behaviors that address key distributed system issues
- … supports standard service specifications
  - Resource mgt, dbms, workflow, security, …
  - Target of current & planned GGF efforts
  - OGSA wg defines "OGSA compliance"
- … and arbitrary application-specific services based on these & other definitions

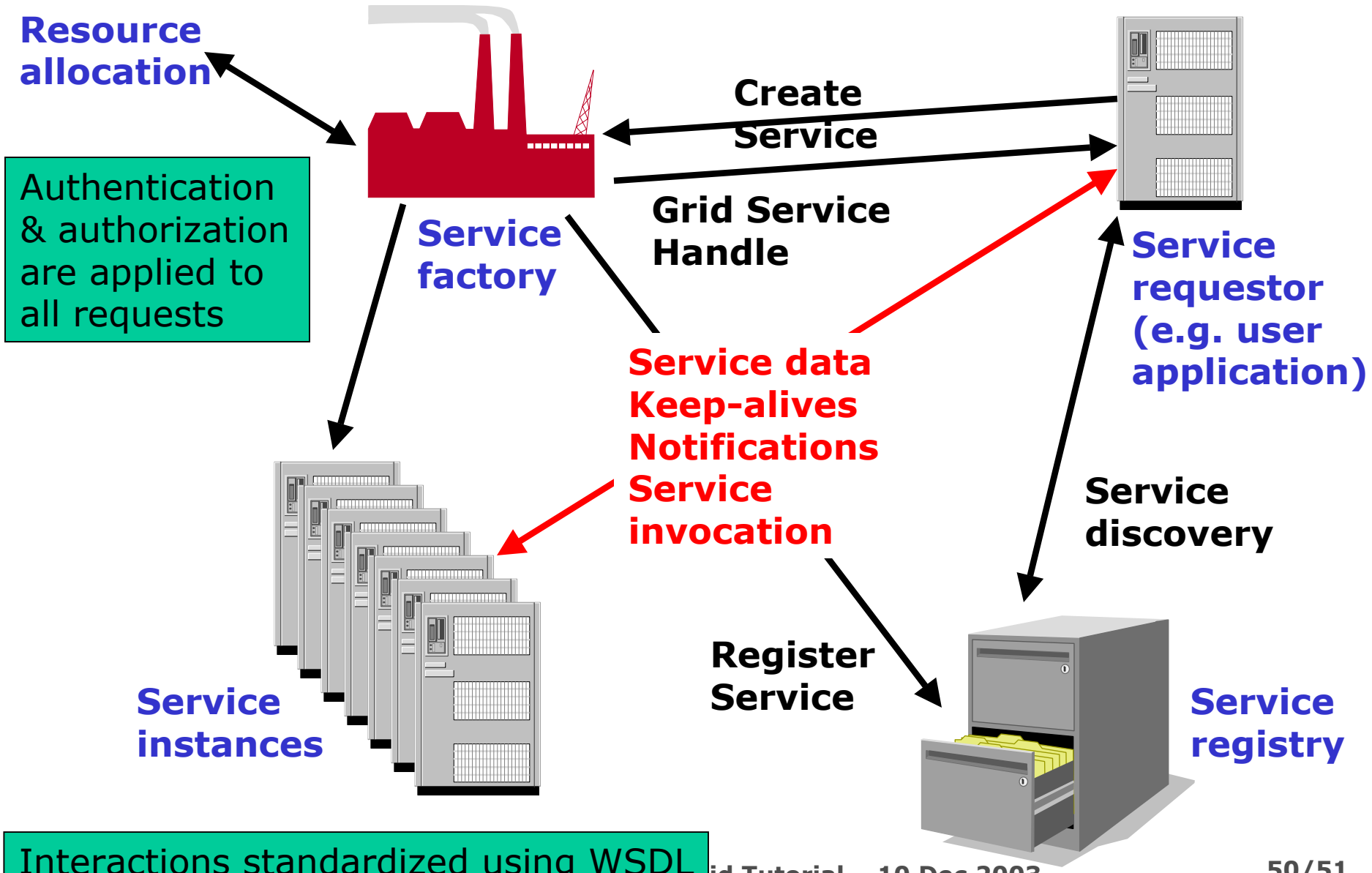
# OGSI Specification

- Defines WSDL conventions and extensions
  - For describing and naming services
  - Working with W3C WSDL working group to drive OGSI extensions into WSDL 1.2
- Defines fundamental interfaces (using extended WSDL) and behaviors that define a Grid Service
  - A unifying framework for interoperability & establishment of total system properties
- http://www.ggf.org/ogsi-wg

# OGSI: Standard Web Services Interfaces & Behaviors

- Naming and bindings (basis for virtualization)
  - Every service instance has a unique name, from which can discover supported bindings
- Lifecycle (basis for fault resilient state management)
  - Service instances created by factories
  - Destroyed explicitly or via soft state
- Information model (basis for monitoring & discovery)
  - Service data (attributes) associated with GS instances
  - Operations for querying and setting this info
  - Asynchronous notification of changes to service date
- Service Groups (basis for registries & collective svcs)
  - Group membership rules & membership management
- Base Fault type

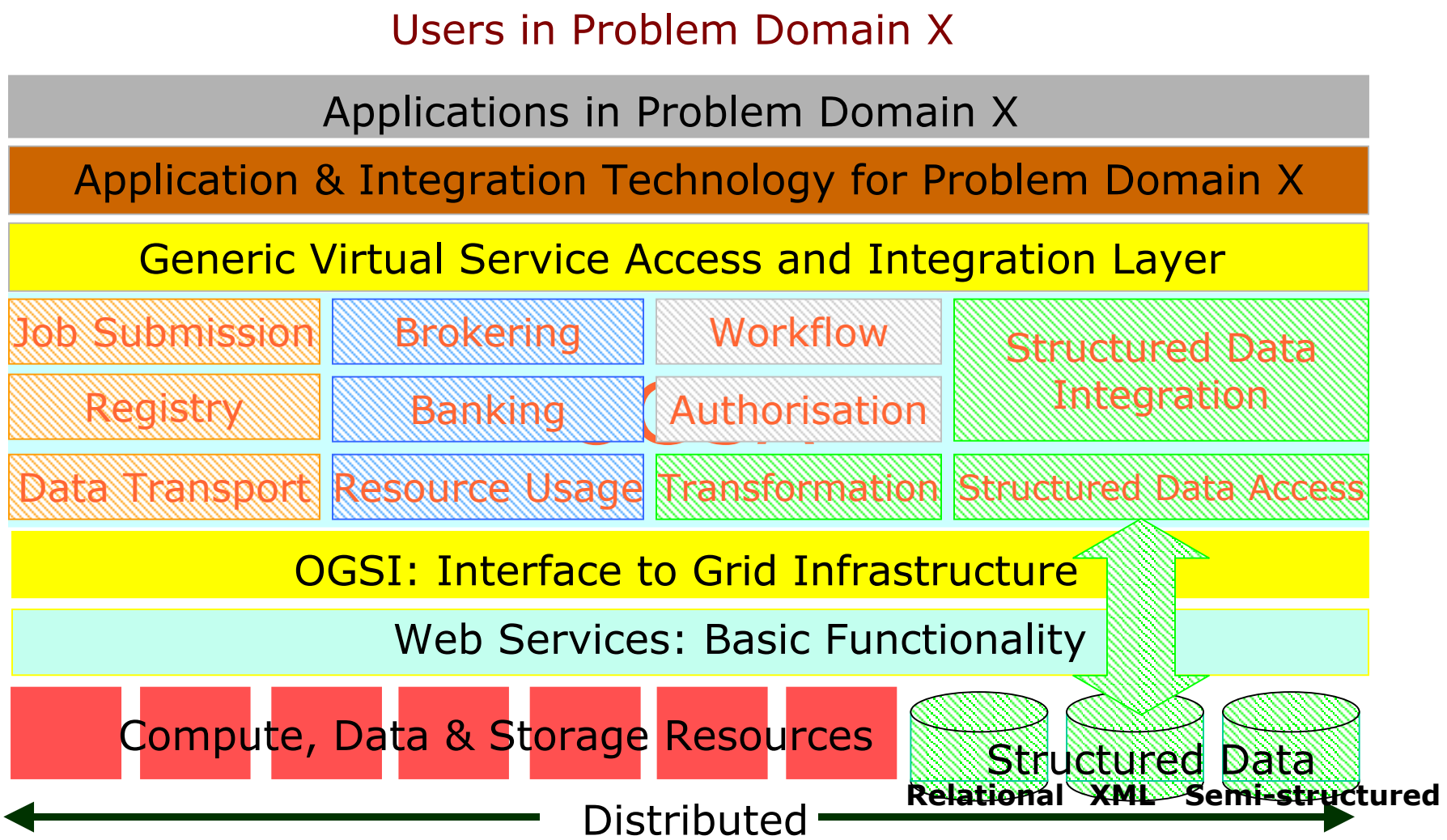
# Open Grid Services Infrastructure



# OGSI Implementations

- Globus Toolkit version 3.0 (Java, C client)
- U Virginia OGSI.NET (.NET)
- LBNL pyGlobus (Python)
- U Edinburgh (.NET)
- U Manchester (PERL)
- Fujitsu Unicore (Java)

# Open Grid Services Architecture




**Virtual Integration Architecture**