Grid Computing 2

Felix Wolf

fwolf@cs.utk.edu

Literature

- Ahmar Abbas: “Grid Computing: A Practical Guide to Technology and Applications”, Charles River Media, 2004
- Fran Berman, Geoffrey G. Fox, Anthony J. G. Hey: Grid Computing – Making the Global Infrastructure a Reality, Wiley, 2003
- Ian Foster, Carl Kesselman: The Grid 2: Blueprint for a New Computing Infrastructure, Morgan Kaufmann, 2003

Outline

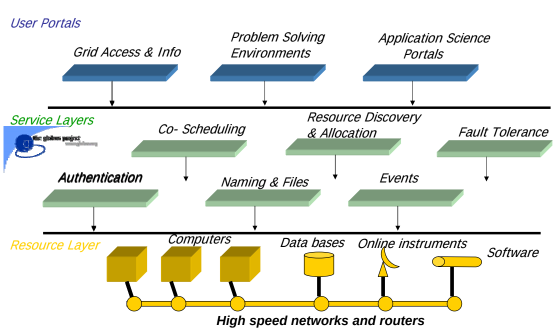
- Grid computing revisited
- Data grids
- Peer-to-peer grids
- Globus
- Open Grid Services Architecture

Grid computing

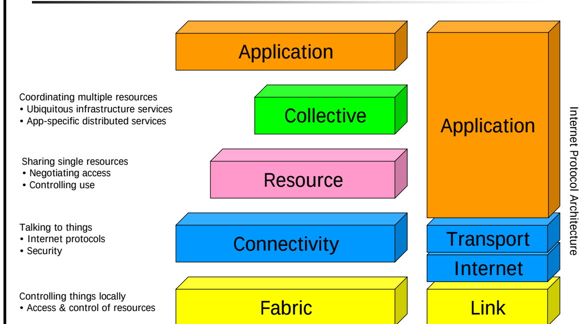
Grid computing enables virtual organizations to share geographically distributed resources as they pursue common goals, assuming the absence of central location, central control, omniscience, and an existing trust relationship.

- Resources
  - Computing resources
  - Databases
  - Visualization facilities
  - Sensors
  - Hand-held devices
  - [...]

Grid architecture



Grid layers



Grid layers

- Application layer
  - Protocols and services targeted toward specific application
  - Least defined in the architecture
- Collective layer
  - System oriented (vs. local) capabilities for wide-scale deployment
  - Index or meta-directory services
  - Resource broker that discover and allocate resources
- Resource layer
  - Grid Resource Allocation Management (GRAM)
  - GridFTP
  - Grid Resource Information Service (GRIS)
- Connectivity layer
  - Core protocols for grid-specific transactions
  - Grid Security Infrastructure (GSI)
- Fabric layer
  - Interface to individual resources (i.e., local resource manager)

Global Grid Forum (GGF)

- Community-initiated forum of thousands of individuals from industry and research
- Leads the global standardization effort for grid computing
- Objectives
  - Promote and support the development, deployment, and implementation of Grid technologies
  - Via the creation and documentation of "best practices“
    - Technical specifications
    - User experiences
    - Implementation guidelines
- Defined “Open Grid Services Infrastructure”

Data grids

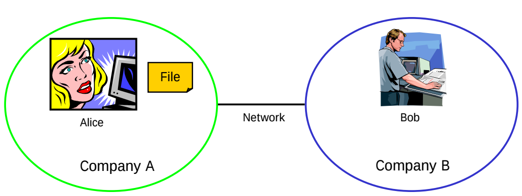
- Introduction
- Alternatives
  - Network File System (NFS)
  - File Transfer Protocol (FTP)
  - NFS over IPSec
  - Secure Copy – scp/sftp
  - De-Militarized Zone (DMZ)
  - GridFTP
  - Andrew File System
- Avaki Data Grid
- Large-scale data grids

Data grids

- Provide transparent, secure, high-performance access to federated data sets across administrative domains and organizations
  - Flat-file data
  - Relational data
  - Streaming data
- Example
  - Collaborators A and B need to share results of computation performed at site A
  - Design data for new part needs to be accessible by team members working on new product at different sites
  - Research community needs access to data generated by large-scale experiment

Scenario

- Alice user on local machine owned by company A
- Bob user on remote machine owned by company B
- Companies do not share a mutually trustful relationship
- Sharing of Alice’s file with Bob has been approved



Network File System (NFS)

- Standard UNIX solution for accessing files on remote machines
  - Remote disk can be made part of local file system
- Bob could run NSF client, Alice could run NSF server
- Advantages
  - Completely transparent
  - Files can be accessed in parts
- Disadvantages
  - Scalability
    - Too many reloads because of short caching period
    - Block size too small
  - Security
    - NSF server has to trust client about UID and GID
  - Requires single identity space
    - Alice and Bob must have accounts on each other’s machines
    - Each one must have same UID on both machines

File Transfer Protocol (FTP)

- Shell-like tool with commands to transfer files
- May be used from script (password in clear-text file)
- Advantages
  - Availability
- Disadvantages
  - Alice needs to remember name and password for Bob's machine
  - Anonymous access may compromise confidentiality
  - Passwords and data transmitted in clear text
    - Firewalls often shut down standard ftp port
  - Consistency
    - After copying, two versions of the same file exist
    - Modification may require new transfer
  - Not conducive to programmatic access

NSF over IPSec

- IETF protocol to encrypt data on a network
- Advantages
  - Transparent encryption of all network traffic including NSF
- Disadvantages
  - Installation requires recompilation of the kernel
    - All machines in Alice's and Bob's domain must be reconfigured
- Does not solve scalability and identity problem

Secure Copy – scp/sftp

- Secure version of rcp/ftp
- scp resembles syntax of UNIX command cp
  - Provision for naming remote user and machine
- Advantages
  - Password and data transfer encrypted
- Disadvantages
  - Alice still needs to remember machine name and password
    - Use of authorized key file possible
  - Does not address consistency problem
  - Programmatic access still problematic

De-Militarized Zone (DMZ)

- Third set of machines accessible to both Alice and Bob
  - Access with scp/sftp
- Advantages
  - Neither party has to grant the other party access to its own machines
- Disadvantages
  - Consistency problem: 3 copies
  - Modification by Alice requires upload and notification of Bob
    - Update essentially requires 2 transfers
    - Increased network traffic
- Scalability
  - New DMZ for every party Alice wants to share a file with

GridFTP

- Tool for transferring files built on top of Globus toolkit
- Advantages
  - Secure communication by encrypting password and data
  - High-performance, concurrent access
  - API allows programmatic access
  - Variety of access patterns: blocked or striped
  - Partial access possible
  - Alice and Bob have to remember only machine name
    - Identities managed by Globus using session-based credentials
- Disadvantages
  - Alice and Bob still need accounts on each other's machines
  - Does not address consistency problem

Andrew File System (AFS)

- Distributed network file system
  - Access to files and directories distributed across multiple sites
- Forms single virtual file system
- Organized as collection of cells
  - Each cell is independently administered file system
  - Alice's and Bob's machines would represent different cells
- Advantages
  - Different cells can be managed by different organizations
    - Alice and wouldn't need accounts on each other's machines
  - Consistency with copy-on-open semantics
  - Intelligent caching
  - Encryption in transit, authentication using Kerberos
- Disadvantages
  - Organizations must change authentication mechanism to Kerberos
  - Organizations must migrate to ASF

Avaki Data Grid

- Successor of Legion (University of Virginia)
  - Start 1993
  - Objective: robust, scalable Grid infrastructure for industrial use
    - Operating system for Grids
    - Object-based design
  - 1997: first deployment, demo at SC'97
- Applied MetaComputing was founded in 1999
  - Objective: transfer of Legion to industry
  - Name was changed to Avaki in 2001
- First commercial release in September 2001
- Avaki is hardened, trimmed-down, focused-on-commercial-requirements version of Legion

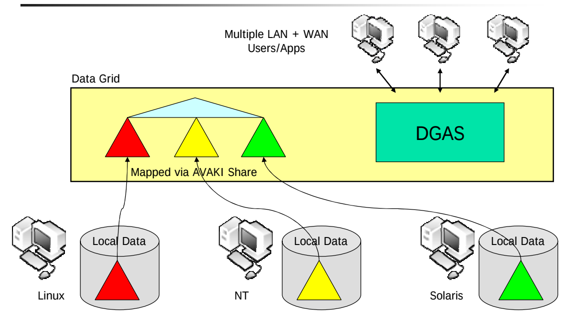
Requirements

- High performance
  - Caching of local copies
- Coherent
  - Cache coherence
  - Exploit different consistency requirements
- Transparent
  - No change of code
- Secure
  - Strong authentication with identities spawning more than one administrative domain and organization
  - Access control and protection of data

Design

- Principles
  - Single-system view
    - No platform-dependent limitations (e.g., file size, portability)
    - Single namespace
  - Transparency as a means of hiding detail
    - Access, location, heterogeneity, failure, migration, replication, scaling, concurrency
    - Grid objects identified by location-independent identifier (LOID)
  - Reduce "activation energy"
- Data Grid consists of a set of DGASs (Data Grid Access Servers) that support the NSF protocol
- Mapping of local directories into Grid via NFS protocol
  - Local file system is NSF client to DGAS
- Access control via access control lists (ACLs)
  - Allow and deny lists
- Implemented in J2EE

Avaki Data Grid



Accessing the data grid

- NFS
  - Normal command-line tools, such as 'ls' still work
- Command line interface
  - Tools, such as 'avaki ls' mimic UNIX commands
  - Only needed if mount not possible
- Web portal
  - Traverse the directory structure
  - Manage access control lists
  - Create and remove shares

Managing the data grid

- Grid is divided into grid domains representing different administrative domains
- Local administrators cooperate to share data
  - Control left in the hands of local owners
- Management tasks
  - Server management
    - Control availability
  - User management
    - Import from LDAP / Active Directory / NIS or defined in Grid itself
  - Object management
    - Creation, deletion, ACL settings
  - Grid interconnects
    - Connections to other Grid domains
  - Grid monitoring

Architecture

- Grid server
  - Domain creation, authentication, access control, metadata management, monitoring searching
  - One distinguished grid server per domain
    - Grid domain controller (GDC)
- Share server
  - Translates grid read read/writes into read/write on underlying file system
  - Bulk transfer, consistency management
- Data grid access server (DGAS)
  - Data access via NFS protocol, caching
  - Access control via signed credentials
- Proxy server
  - Single point of contact behind firewall for each grid domain
  - Transparent en-/decryption, / (de)compression
- Fail-over server
  - Backup for GDC
  - In case of GDC failure, limited set of actions to ensure consistency

Large Hadron Collider at CERN

- Will collide beams of protons at an energy of 14 TeV
- Most powerful device of its kind (17 mile ring)
- Will allow scientists to recreate the conditions just after the "Big Bang"
- Multinational effort
  - 10,000 scientists from 50 nations involved world wide
- Expected to be operational in 2007
- Location: France and Switzerland near Geneva
- CERN = European Organization for Nuclear Research

1 TeV is about the energy of motion of a flying mosquito. What makes the LHC so extraordinary is that it squeezes energy into a space about a million million times smaller than a mosquito.

Data and computing requirements

- More than 10 Petabytes of experimental data annually
- Grid technology needed to support analysis chain from first recording down to the scientist's desktop
- Grid follows multi-tier hierarchical model
  - Tier 0 (based at CERN)
    - Raw data storage, first-level reconstruction, event summaries
  - Tier 1 (several regional centers)
    - Analysis and Monte-Carlo data generation
    - Partial data storage of event summary data and user support
  - Tier 2 (national centers of lower capacity)
    - Participate in Monte-Carlo production
    - Produce and store Analysis Object Data and Derived Physics Data
  - Tier 3 and 4
    - Support university group and end user workstation

Peer-to-peer grids

- Resource to be shared is the storage capacity of individual desktop workstations
- Previously file-sharing systems were largely confined to
  - Local Area Networks (LANs)
  - Exchange of files with know individuals over the Internet
- Advanced P2P systems follow two models
  - Centralized approach (Napster)
  - Decentralized approach (Gnutella)
- Efficient file sharing
  - Bit Torrent

Centralized approach

- Central server system
  - Directs traffic between individual registered users
  - Maintains directories of shared files stored on client PCs
  - Update every time a user logs on / off
- If user requests a file
  - Server creates list of matching files by cross-checking with the database of currently connected users
  - List is displayed to user
  - User makes selection and directly downloads file from machine where file is located
  - Actual file is never stored at central server or intermediate point on the network
- Example: Napster

Decentralized approach

- No central server to keep track of files
- Relies on each individual computer to announce its existence to a peer
- The peer forwards announcement to all other peers it is connected to
- Requests are handled in a similar fashion
  - Location of requested file is forwarded through chain of peers
  - Direct connection is established between requester and owner to perform the actual download
- Example: Gnutella

Efficient file sharing

- Problem: publishing a popular file
  - All upload cost is placed on the hosting machine
- Bit Torrent: cooperative downloading
  - While a client is downloading, it simultaneously uploads pieces of the file to other clients
  - Supporting a large number of down-loaders using only limited resources
  - The more one provides to others the better treatment it gets
    - "Give and ye shall receive!"
- Tracker helps peers to find each other
  - Runs on the server side
  - Keeps track of clients
  - Gives clients random lists of peers downloading at the same time

Globus

- Overview
- History
- Overview papers
- Services

the globus toolkit™
www.globustoolkit.org

- http://www.globus.org

Overview

- Open source software toolkit used for building grids
- Developed by the Globus Alliance
- Allows users to share resources securely online
  - Across corporate, institutional, and geographic boundaries
  - Without sacrificing local autonomy
- Includes software for
  - Security
  - Data management
  - Resource management
  - Information infrastructure
- Globus version 3 based on new open-standard Grid services

History

- The Globus project was created in 1994
- Roots go back to 1992
  - Ian Foster, Carl Kesselman, and Steve Tuecke worked on a mutli-method communication library called Nexus
- Proposal to DARPA submitted in 1994
  - "a platform for constructing parallel and distributed computations in large-scale internetworked environments"
  - Nexus became Globus
- I-WAY showcase at SC'95
  - Considered start of "Grid" computing
  - 17 sites connected sites to create single virtual supercomputer
- DARPA funds Globus project in 1996
- Key players were
  - Information Sciences Institute at University of California
  - University of Chicago
  - Argonne National Laboratory

Overview papers

- Ian Foster, Carl Kesselman, Steven Tuecke: "The Anatomy of the Grid – Enabling Scalable Virtual Organizations", International Journal of High Performance Computing Applications, 15(3), 2001.
  - Defines Grid computing
  - Proposes a Grid architecture
- Ian Foster, Carl Kesselman, Jeffrey M. Nick, Steven Tuecke: "The Physiology of the Grid – An Open Grid Services Architecture for Distributed Systems Integration", Open Grid Service Infrastructure WG, Global Grid Forum, June 2002.
  - Describes how Grid functionality can be incorporated into a Web Services framework
  - Introduces OGSA

Grid Security Infrastructure (GSI)

- Secure communication
  - Between elements of a computation Grid
  - Authenticated and perhaps confidential
- Security across organizational boundaries
  - Without centrally-managed security system
- Single sign-on
  - Including delegation of credentials for computations involving multiple resources and / or sites

Public-key cryptography

- Problem with symmetric (traditional) cryptography
  - Sender and receiver use the same key to de/encrypt data
  - Problem: key exchange
- PKC relies on two keys to encode and decode messages
  - One private and one public
  - Critical that private key is kept private
- Keys are mathematically related numbers
  - If one key is used to encrypt data, the other can be used to decrypt it
- Next to impossible to derive one key from the other
  - Requires e.g. factoring a large number
- Application
  - Privacy / integrity of data, mutual authentication, single sign-on

Digital signatures

- Purpose: making sure that message is authentic
  - Has not been modified by someone else
- To digitally sign a document
  - Compute hash of the document
    - Hash is condensed version of the document
    - Algorithm must be known to recipient – but not a secret
  - Encrypt hash using your private key and attach to message
- To verify that document is authentic
  - Compute hash of the document
  - Decrypt sender's hash that came along with the message
  - Document is authentic if both hash values match

Certificates

- In GSI every user and service is identified via a certificate
- A GSI certificate contains the following information
  - A subject name identifying the person or object
  - The public key belonging to the subject
  - The certificate authority (CA) that has signed the certificate
  - The digital signature of the named CA
- CA is used to verify the link between the subject and its public key
  - CA must be trusted
- Certificate stored in X.509 format (IETF standard)

Mutual authentication

- Requirement
  - Both parties A and B have certificates
  - They trust the CA's that have signed each others certificates
    - Must have copies of the CA's certificates
    - Must trust that certificates belong to CAs
- GSI uses SSL for mutual authentication
  - A gives B her certificate
  - B validates A's certificate by checking CA's signature
  - B sends random message to A
  - A encrypts random message using private key and returns it
  - B decrypts message using A's public key and compares to original message (must match)
  - Same operation in reverse

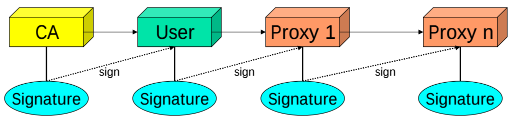
Confidential communication

- By default, GSI does not encrypt communication
  - Avoids overhead
- GSI can be easily used to establish shared (symmetric) key for encryption
- Communication integrity is established by default
  - Eavesdropper may listen to but not modify traffic
  - Overhead small compared to encryption

Delegation and single sign-on

- Securing private keys
  - File with private key is encrypted using a passphrase
  - To use GSI user must enter passphrase
- Problem
  - Grid computations involving several resources (each of which requiring mutual authentication) would require the user to enter the passphrase multiple times
  - Sometimes agents (local or remote) need to request services on the user's behalf
- GSI provides delegation capability
  - Extension of standard SSL protocol to reduce the number of times the user must enter the passphrase
  - Involves creation of a proxy

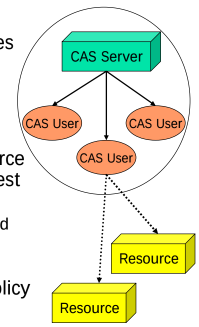
Proxy



- New certificate with new public key in it
  - Signed by owner instead of CA
  - Includes expiration time (proxies have limited lifetime)
- Private key protected by normal access control
  - Limited security requirements because of short lifetime
- Mutual authentication
  - Remote party receives proxy certificate plus owner certificate
  - Chain of trust from CA to the proxy (through the owner)

Community Authorization Service (CAS)

- Allows resource providers to specify course-grained access control policies in terms of communities as a whole
- Fine-grained access control policy management is delegated to the community itself
- A user who wants to access a resource served by a CAS server makes request to that CAS server
  - Server returns GSI proxy with embedded policy
  - User contacts resource using the proxy
- Resource applies local policy plus policy embedded in proxy



GridFTP

- Motivation: access to distributed data as important as access to distributed computational resources
  - Transfer of large amounts of data between storage systems
  - Access to large amounts of data by distributed applications and users
- GridFTP is high performance, secure, reliable data transfer protocol
  - Based on FTP
- Features
  - GSI security on control and data channels
  - Multiple data channels for parallel transfers
  - Partial file transfers
  - Third-party transfers (direct server-to-server)
  - Authenticated data channels
  - Reusable data channels
  - Command pipelining

Reliable File Transfer (RFT)

- Problem
  - High-volume data transfer that needs many days to complete
  - GridFTP requires the client to keep a control channel
    - TCP/IP socket open to participating servers
  - Client must stay on the net without any problems
    - NSF failure or network problems?
    - How to deal with mobile clients?
  - GridFTP is robust with respect to remote failures
    - If client fails, state of transfer is lost
    - Where to resume transfer?
- RTF is batch GridFTP with Web Service interface (OGSI)
- Will be WSRF compliant with GT 4

Replica Location Service (RLS)

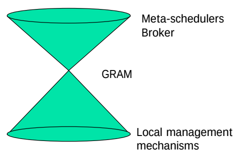
- Replication of data items can
  - Reduce access latency
  - Improve data locality
  - Increase robustness, scalability, and performance
- RLS manages mapping information
  - Between logical names for data items and target names
- RLS usually component of a data-grid architecture
  - Other components include reliable file transfer, metadata management, reliable replication, and workflow management

XIO

- Extensible input / output library for GT
  - Simple API (open/close/read/write)
- Objectives
  1. Provide single user API to all Grid IO protocols
     - Simplifies change to alternate protocol
     - New protocols can be added in the form of drivers
  2. Minimize time to create/prototype new protocols
     - Provides driver interface
     - Takes responsibility for
       - Error checking
       - Asynchronous message delivery
       - Timeouts

GRAM

- Grid Resource Allocation and Management
- Single standard interface for requesting and and using remote system resources for the execution of jobs
- Typical use: remote job submission and control
- Supports distributed computing applications
- Single protocol and API
- Uniform interface to local job management systems
  - Schedulers
  - Queuing systems
  - Reservations systems



Index Service

- Each Grid service instance has a set of service data associated with it
- The Index Service provides an interface for
  - Generating service data
  - Aggregating service data
  - Querying service data
- Can be used as registry of Grid services
  - Registration of a set of services (service group)
  - Periodic updates
  - Query and service data aggregation

Open Grid Services Architecture (OGSA)

- Service-oriented architecture
- Web Services
- OGSA Overview
- Open Grid Services Infrastructure (OGSI)
- Implementing Grids services
- Web Services Resource Framework

Service-oriented architecture (SOA)

- Service is a function that can be invoked via a well-defined remote interface
- A system defined in terms of services allows independent development and management of client and provider
- SOA advocates looser coupling among interacting software systems compared to earlier forms of distributed computing
  - RPC, Java RMI, CORBA, DCOM
- Service defined in terms of request/reply messages
  - Behavior may depend on state (files, databases, sensors, ...)
- Client relies on protocol
- Good service implementation supports multiple protocols
  - E.g., http and https

Web services

- Motivation
  - Application integration across the Internet
- Problem with traditional communication middleware
  - E.g., CORBA, Java RMI, DCOM
  - Doesn't support heterogeneity (platforms and languages)
  - Doesn't work across the Internet
  - Isn't pervasive
  - Hard to use
  - Expensive
  - High maintenance costs
- Web services represent new form of middleware based on XML and the Web
  - Language and platform neutral

Web services (2)

- A Web service is a Web resource
  - Can be accessed using platform- and language neutral Web protocols, such as HTTP
- A Web service provides an interface (Web API)
  - Can be called from another program
  - E.g., Google Web API
  - Described in a WSDL document
- A Web service is typically registered and can be located through a registry (UDDI)
- Web services support loosely coupled connections between systems
  - Communication through exchange of XML (SOAP) messages

Extensible Markup Language (XML)

- Structured way to represent information
  - W3C Recommendation (http://www.w3c.org)
- XML document is hierarchy of XML elements
- Elements consist of markup and content
- XML is meta-markup language
  - No fixed set of tags
- Applications can use XML parser to read XML documents

```
<?xml version 1.0?>
<product barcode="2394287410">
  <manufacturer>Verbatim</manufacturer>
  <name>Data Life MF 2HD</name>
  <size>3.5"</size>
  <color>black</color>
  <description>floppy</description>
</product>
```

XML Schema

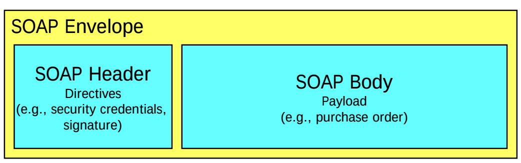
- XML document specifying kind and structure of elements
  - E.g., structure of purchase order document
- Instance documents
  - XML documents described by schema
- Schema-valid
  - XML documents satisfying all constraints specified in the schema
- Schema vs. Document Type Definition (DTD)
  - Simple and complex data types
  - Type derivation and inheritance
  - Element occurrence constraints

```
<?xml version 1.0?>
<xs:schema xmlns:xs="http://www.w3c.org/2001/XMLSchema">
  <xs:element name="fullName" type="xs:string/>
</xs:schema>
```

```
<?xml version 1.0?>
<fullName>Alan Turing</fullName>
```

Simple Object Access Protocol (SOAP)

- XML protocol used to communicate with Web services
- Follows stateless, one-way paradigm
  - Applications can create more complex interaction patterns
- Consists of an envelope that contains header and body
- Header includes control information
- Body includes actual data
- Can be transported using different protocols (e.g., HTTP, SMTP)



Web Services Description Language (WSDL)

- XML format for describing Web services as a set of endpoints operating on messages
- Operations and messages are described abstractly
  - Later bound to a concrete network protocol
- 6 major components (WSDL 1.1)
  - Types
    - XMLSchema data types used to define messages
  - Message
    - Definition of the data being transmitted (i.e., one-way message)
  - PortType
    - Combines multiple messages to form operation (e.g., input and output message)
  - Binding
    - Protocol and data format (e.g., SOAP HTTP)
  - Service
    - Defines the address for invoking the service (i.e., URL)

UDDI

- OASIS standard
  - Universal Description, Discovery, and Integration
- Specifies
  - Protocols for creating Web service registries
  - Methods for controlling access to the registry
  - Mechanism for distributing records to other registries
- UDDI is itself a Web service
- Manages information about service types and providers
- White pages offer provider identity information
- Yellow pages help find provider for given service type
  - Search along taxonomies
- Green pages provide technical information (e.g., WSDL)
- UDDI is mostly used at development time
  - Can be used at runtime to generate dynamic proxy

OGSA Overview

- Defines infrastructure for integrating and managing services within a virtual organization
- Architecture for building Grid applications
- Introduced in "Physiology" paper
- Views Grid as an extensible set of Grid services
- Aligns Grid technology with Web services
  - Extends notion of a Web service
- Architecture consists of a set of basic interfaces
  - Can be used to build Grids
- Open
  - Vendor neutral, extensible, community driven

Open Grid Services Infrastructure (OGSI)

- Everything is a Grid service
- Grid services can be
  - Permanent or transient
  - Long or short lived
- Each Grid service supports specific set of interfaces
  - With well-defined behavior
- OGSI extended WSDL to GSDL
  - Inheritance of Web service interfaces (portTypes)
    - New portTypes can be constructed by extending an existing one
    - Addressed in WSDL 2.0 (Working Draft)
  - Service Data Declarations
    - Help describe identity, interfaces, and state of the service

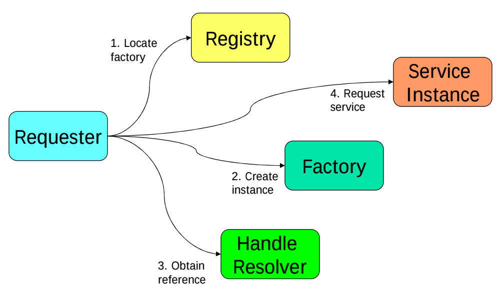
Grid services

- Extends portType ogsi:GridService
  - Inherits service data declarations
  - Inherits operations
- Created by factory
  - Client can specify lifetime
- Identity
  - Grid service handle (GSH) uniquely identifies the service instance
  - Grid service reference (GSR) includes binding (e.g., WSDL)
    - Obtained by passing handle to handle resolver service
- State described by service data elements
  - Generic operations to get and set values
  - Interfaces available to receive notification of changes
  - Service data elements may have lifetimes

The GridService portType

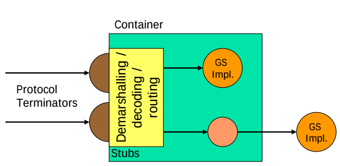
- Service data elements
  - ServiceDataName
    - All supported SDEs
  - Interface
    - All supported portTypes
  - factoryLocator
    - Reference to factory that has created the service
  - gridServiceHandle
  - gridServiceReference
    - One value must be a WSDL representation
  - terminationTime
  - [...]
- Grid service operations
  - find- / setServiceData
  - Destroy
  - RequestTerminationBefore
  - RequestTerminationAfter

Basic services and their roles



Implementing Grid services

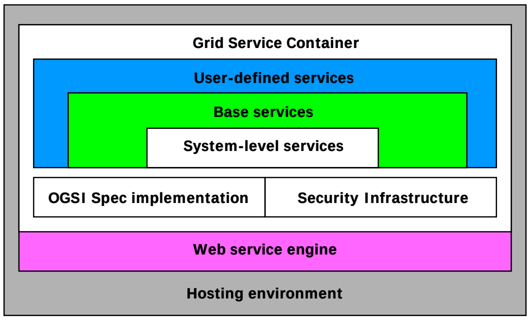
- Server-side programming pattern
  - Monolithic: all in one executable
  - Container (e.g., J2EE)



Web Services Resource Framework

- “State” appears in almost all applications
- Web services might model state in different ways
  - OGSI v1.0 defined one approach
  - WS-Resource Framework proposes an evolution of that approach
  - Ad-hoc approaches
- Standardization encourages tooling & code re-use
- Previous Web service standards not sufficient
- Concerns about OGSI
  - Too much in one specification
  - Does not work well with existing Web services tooling
  - Too object oriented
- WSRF (submitted to OASIS)
  - Splits OGIS into family of specifications
  - Tones down usage of XML Schema
  - Explicit distinction between service and “stateful” resource

GT 3 Core architecture



GT 3 Core architecture (cont.)

- OGSI Spec implementation
  - Implementation of all OGSI-specified interfaces
- Security infrastructure
  - Session and per-message security (GSI)
- System-level service
  - Generic enough to be used by other Grid services
  - Ping service
  - Logging management service (log filters)
  - Management service (monitoring and shut down)
- Base services
  - Higher-level services, such as GRAM or RFT
- User-defined services
  - Services outside the Globus toolkit