Grid Computing

1

The Grid Problem

- Flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resource
  - From "The Anatomy of the Grid: Enabling Scalable Virtual Organizations"
- Enable communities ("virtual organizations") to share geographically distributed resources as they pursue common goals -- assuming the absence of…
  - central location,
  - central control,
  - omniscience,
  - existing trust relationships.

2

Broader Context

- "Grid Computing" has much in common with major industrial thrusts
  - Business-to-business, Peer-to-peer, Application Service Providers, Storage Service Providers, Distributed Computing, Internet Computing…
- Sharing issues not adequately addressed by existing technologies
  - Complicated requirements: "run program X at site Y subject to community policy P, providing access to data at Z according to policy Q"
  - High performance: unique demands of advanced & high-performance systems

3

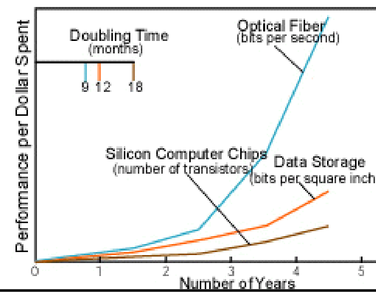
Why Now?

- Moore's law improvements in computing produce highly functional endsystems
- The Internet and burgeoning wired and wireless provide universal connectivity
- Changing modes of working and problem solving emphasize teamwork, computation
- Network exponentials produce dramatic changes in geometry and geography

4

Network Exponentials

- Network vs. computer performance
  - Computer speed doubles every 18 months
  - Network speed doubles every 9 months
  - Difference = order of magnitude per 5 years
- 1986 to 2000
  - Computers: x 500
  - Networks: x 340,000
- 2001 to 2010
  - Computers: x 60
  - Networks: x 4000



5

The Globus Project™

Making Grid computing a reality

- Close collaboration with real Grid projects in science and industry
- Development and promotion of standard Grid protocols to enable interoperability and shared infrastructure
- Development and promotion of standard Grid software APIs and SDKs to enable portability and code sharing
- The Globus Toolkit™: Open source, reference software base for building grid infrastructure and applications
- Global Grid Forum: Development of standard protocols and APIs for Grid computing

6

Some Important Definitions

- Resource
- Network protocol
- Network enabled service
- Application Programmer Interface (API)
- Software Development Kit (SDK)

7

Resource

- An entity that is to be shared
  - E.g., computers, storage, data, software
- Does not have to be a physical entity
  - E.g., Condor pool, distributed file system, …
- Defined in terms of interfaces, not devices
  - Open/close/read/write define access to a distributed file system, e.g. NFS, AFS, DFS

8

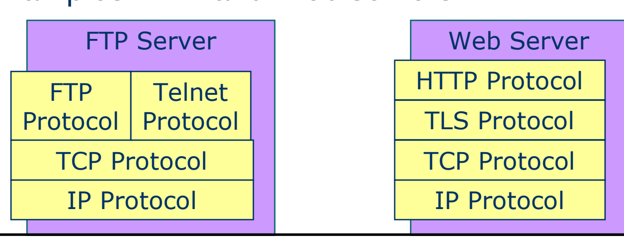
Network Protocol

- A formal description of message formats and a set of rules for message exchange
  - Rules may define sequence of message exchanges
  - Protocol may define state-change in endpoint, e.g., file system state change
- Good protocols designed to do one thing
  - Protocols can be layered
- Examples of protocols
  - IP, TCP, TLS (was SSL), HTTP, Kerberos

9

Network Enabled Services

- Implementation of a protocol that defines a set of capabilities
  - Protocol defines interaction with service
  - All services require protocols
  - Not all protocols are used to provide services (e.g. IP, TLS)
- Examples: FTP and Web servers



10

Application Programming Interface

- A specification for a set of routines to facilitate application development
  - Refers to definition, not implementation
  - E.g., there are many implementations of MPI
- Spec often language-specific (or IDL)
  - Routine name, number, order and type of arguments; mapping to language constructs
  - Behavior or function of routine
- Examples
  - GSS API (security), MPI (message passing)

11

Software Development Kit

- A particular instantiation of an API
- SDK consists of libraries and tools
  - Provides implementation of API specification
- Can have multiple SDKs for an API
- Examples of SDKs
  - MPICH, Motif Widgets

12

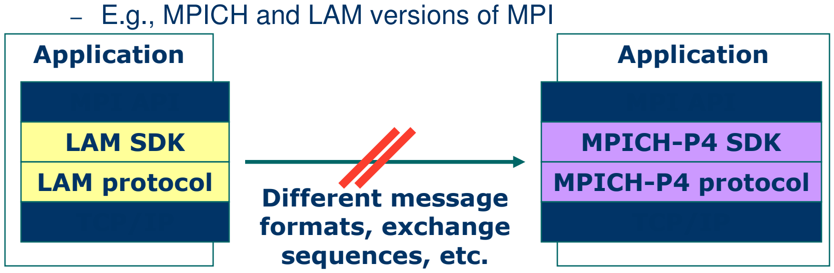
A Protocol can have Multiple APIs

- TCP/IP APIs include BSD sockets, Winsock, System V streams, …
- The protocol provides interoperability: programs using different APIs can exchange information
- I don't need to know remote user's API

13

An API can have Multiple Protocols

- MPI provides portability: any correct program compiles & runs on a platform
- Does not provide interoperability: all processes must link against same SDK
  - E.g., MPICH and LAM versions of MPI



14

APIs and Protocols are Both Important

- Standard APIs/SDKs are important
  - They enable application portability
  - But w/o standard protocols, interoperability is hard (every SDK speaks every protocol?)
- Standard protocols are important
  - Enable cross-site interoperability
  - Enable shared infrastructure
  - But w/o standard APIs/SDKs, application portability is hard (different platforms access protocols in different ways)

15

Grid Architecture

16

Why Discuss Architecture?

- Descriptive
  - Provide a common vocabulary for use when describing Grid systems
- Guidance
  - Identify key areas in which services are required
- Prescriptive
  - Define standard "Intergrid" protocols and APIs to facilitate creation of interoperable Grid systems and portable applications

17

One View of Requirements

- Identity & authentication
- Authorization & policy
- Resource discovery
- Resource characterization
- Resource allocation
- (Co-)reservation, workflow
- Distributed algorithms
- Remote data access
- High-speed data transfer
- Performance guarantees
- Monitoring
- Adaptation
- Intrusion detection
- Resource management
- Accounting & payment
- Fault management
- System evolution
- Etc.
- Etc.
- …

18

Another View: "Three Obstacles to Making Grid Computing Routine"

- New approaches to problem solving
  - Data Grids, distributed computing, peer-to-peer, collaboration grids, …
- Structuring and writing programs
  - Abstractions, tools
- Enabling resource sharing across distinct institutions
  - Resource discovery, access, reservation, allocation; authentication, authorization, policy; communication; fault detection and notification; …

19

Programming & Systems Problems

- The programming problem
  - Facilitate development of sophisticated apps
  - Facilitate code sharing
  - Requires programming environments
    - APIs, SDKs, tools
- The systems problem
  - Facilitate coordinated use of diverse resources
  - Facilitate infrastructure sharing
    - e.g., certificate authorities, information services
  - Requires systems
    - protocols, services

20

The Systems Problem: Resource Sharing Mechanisms That …

- Address security and policy concerns of resource owners and users
- Are flexible enough to deal with many resource types and sharing modalities
- Scale to large number of resources, many participants, many program components
- Operate efficiently when dealing with large amounts of data & computation

21

Aspects of the Systems Problem

- Need for interoperability when different groups want to share resources
  - Diverse components, policies, mechanisms
  - E.g., standard notions of identity, means of communication, resource descriptions
- Need for shared infrastructure services to avoid repeated development, installation
  - E.g., one port/service/protocol for remote access to computing, not one per tool/appln
  - E.g., Certificate Authorities: expensive to run
- A common need for protocols & services

22

Hence, a Protocol-Oriented View of Grid Architecture, that Emphasizes …

- Development of Grid protocols & services
  - Protocol-mediated access to remote resources
  - New services: e.g., resource brokering
  - "On the Grid" = speak Intergrid protocols
  - Mostly (extensions to) existing protocols
- Development of Grid APIs & SDKs
  - Interfaces to Grid protocols & services
  - Facilitate application development by supplying higher-level abstractions
- The (hugely successful) model is the Internet

23

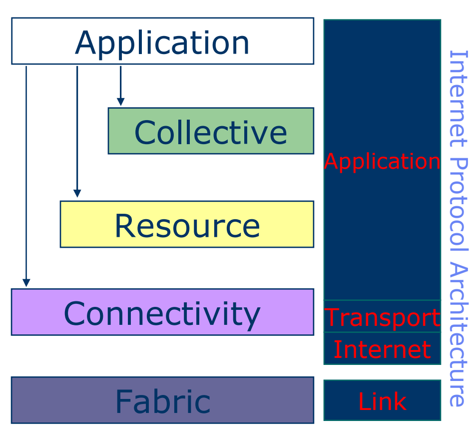
Layered Grid Architecture (By Analogy to Internet Architecture)

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

"Sharing single resources": negotiating access, controlling use

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

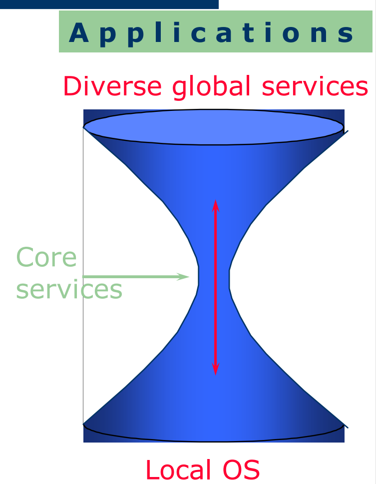
"Controlling things locally": Access to, & control of, resources



24

The Hourglass Model

- Focus on architecture issues
  - Propose set of core services as basic infrastructure
  - Use to construct high-level, domain-specific solutions
- Design principles
  - Keep participation cost low
  - Enable local control
  - Support for adaptation
  - "IP hourglass" model



25

Where Are We With Architecture?

- No "official" standards exist
- But:
  - Globus Toolkit™ has emerged as the de facto standard for several important Connectivity, Resource, and Collective protocols
  - GGF has an architecture working group
  - Technical specifications are being developed for architecture elements: e.g., security, data, resource management, information
  - Internet drafts submitted in security area

26

Fabric Layer Protocols & Services

- Just what you would expect: the diverse mix of resources that may be shared
  - Individual computers, Condor pools, file systems, archives, metadata catalogs, networks, sensors, etc., etc.
- Few constraints on low-level technology: connectivity and resource level protocols form the "neck in the hourglass"
- Defined by interfaces not physical characteristics

27

Connectivity Layer Protocols & Services

- Communication
  - Internet protocols: IP, DNS, routing, etc.
- Security: Grid Security Infrastructure (GSI)
  - Uniform authentication, authorization, and message protection mechanisms in multi-institutional setting
  - Single sign-on, delegation, identity mapping
  - Public key technology, SSL, X.509, GSS-API
  - Supporting infrastructure: Certificate Authorities, certificate & key management, …

GSI: www.gridforum.org/security/gsi

28

Resource Layer Protocols & Services

- Grid Resource Allocation Management (GRAM)
  - Remote allocation, reservation, monitoring, control of compute resources
- GridFTP protocol (FTP extensions)
  - High-performance data access & transport
- Grid Resource Information Service (GRIS)
  - Access to structure & state information
- Others emerging: Catalog access, code repository access, accounting, etc.
- All built on connectivity layer: GSI & IP

GRAM, GridFTP, GRIS: www.globus.org

29

Collective Layer Protocols & Services

- Index servers aka metadirectory services
  - Custom views on dynamic resource collections assembled by a community
- Resource brokers (e.g., Condor Matchmaker)
  - Resource discovery and allocation
- Replica catalogs
- Replication services
- Co-reservation and co-allocation services
- Workflow management services
- Etc.

Condor: www.cs.wisc.edu/condor

30

Summary

- The Grid problem: Resource sharing & coordinated problem solving in dynamic, multi-institutional virtual organizations
- Grid architecture emphasizes systems problem
  - Protocols & services, to facilitate interoperability and shared infrastructure services
- Globus Toolkit™: APIs, SDKs, and tools which implement Grid protocols & services
  - Provides basic software infrastructure for suite of tools addressing the programming problem

31