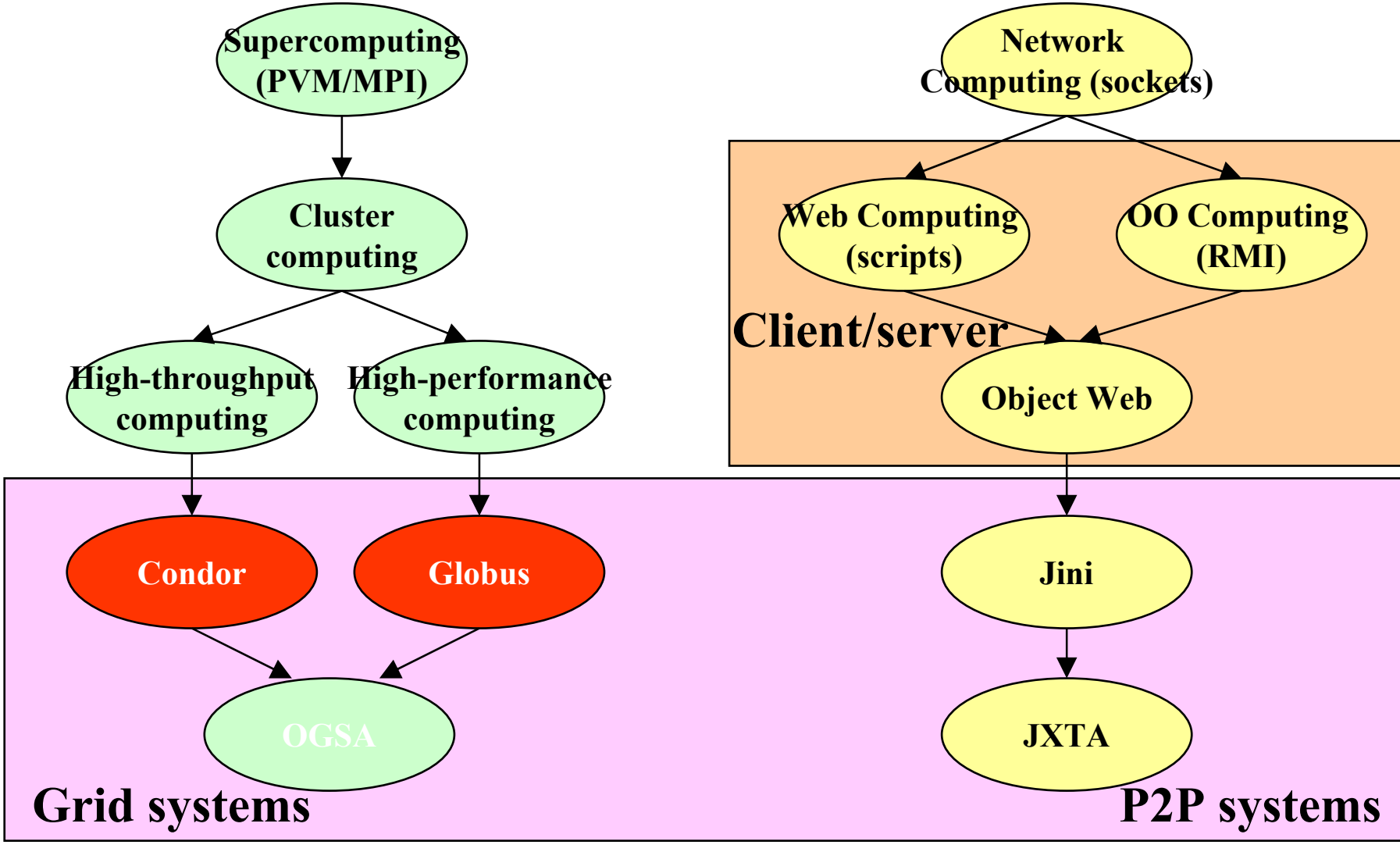
Globus Toolkit

Introduction

The Globus Project Team

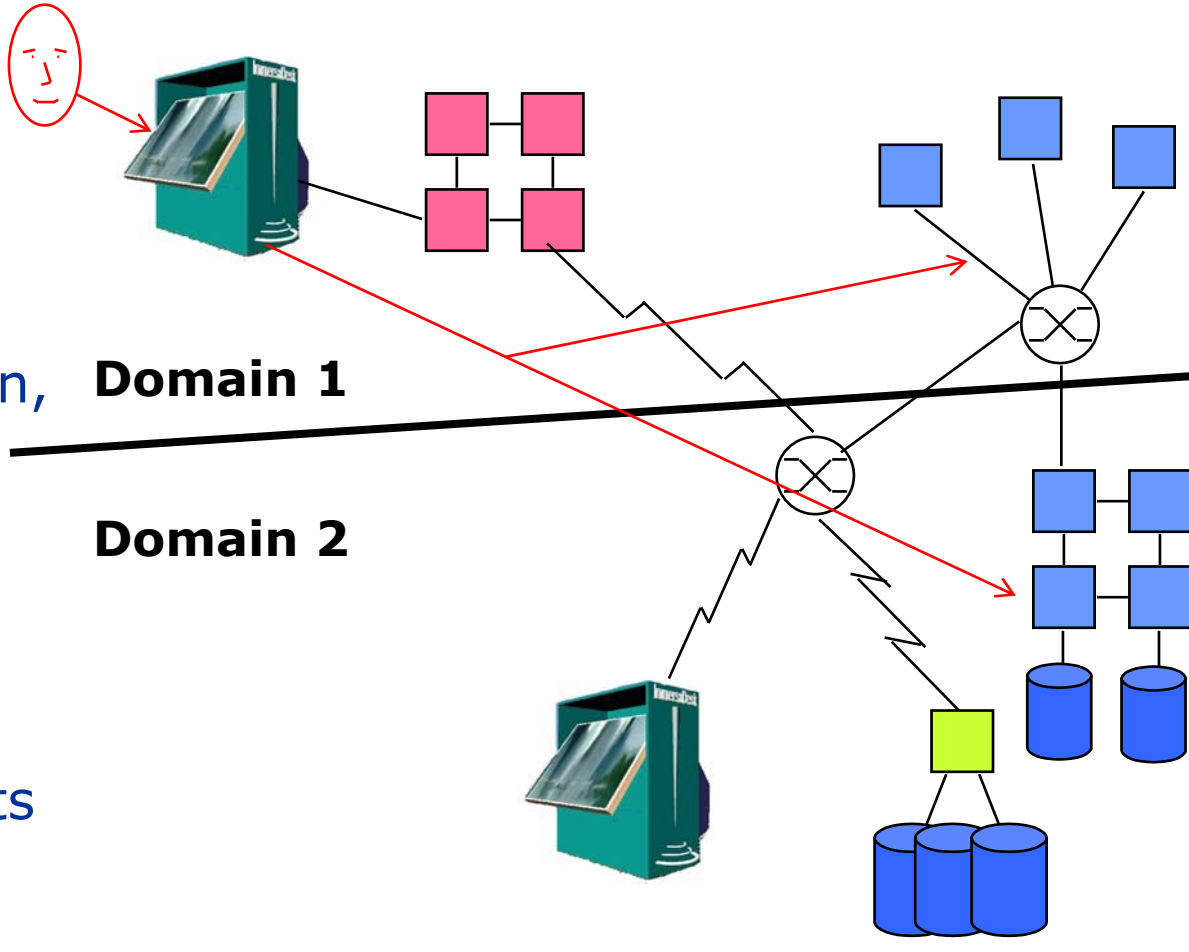
http://www.globus.org

Progress to Grid and P2P systems



Issues

- Authenticate once
- Specify simulation (code, resources, etc.)
- Locate resources
- Negotiate authorization, acceptable use, etc.
- Acquire resources
- Initiate computation
- Steer computation
- Access remote datasets
- Collaborate on results
- Account for usage



Architectural Approaches

- Distributed systems: DCE, CORBA, Jini, etc.
  - Rich functionality eases app development
  - Complexity hinders deployment
    - especially in absence of global control
  - Performance difficulties
- Internet/Web Protocols and Tools
  - Simple protocols facilitate deployment
  - Missing functionality hinders app development
  - Performance difficulties

Standards & Commodity Tech

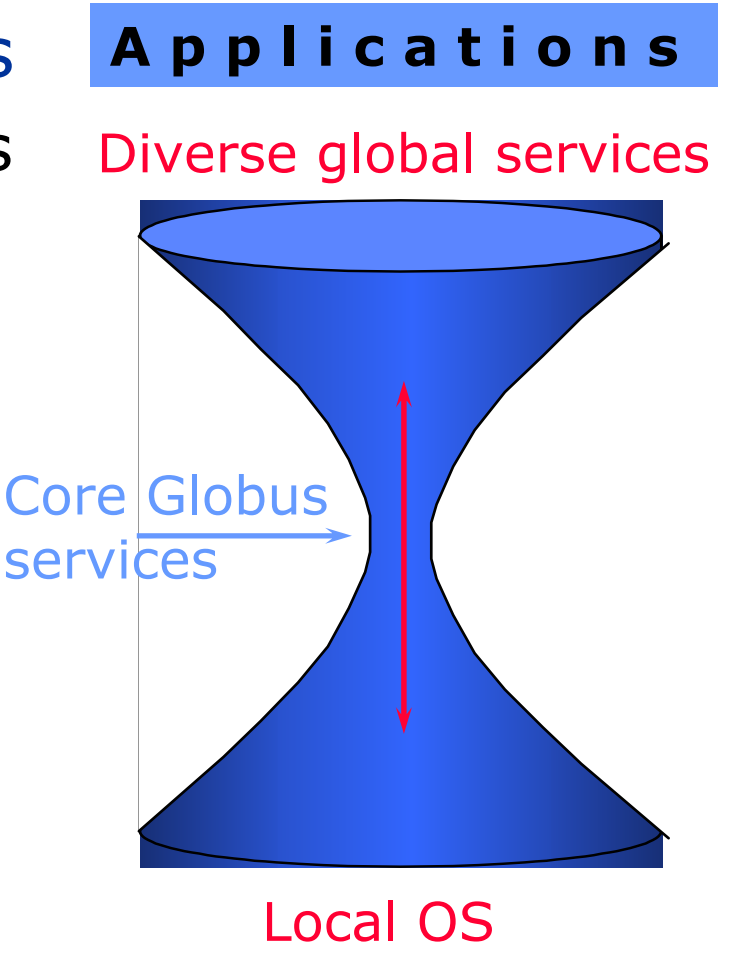
- Where appropriate, exploit standards and commodity technology in core infrastructure
  - LDAP, SSL/TLS, X.509, GSS-API, http, ftp, XML, SOAP, etc.
  - Provides leverage
- Interface with other common standards
  - CORBA, Java/Jini, DCOM, Web, etc
  - While our core infrastructure may not be built on one of these distributed architectures, we can and must cleanly interface with them

The Globus Project

- Basic research in grid-related technologies
  - Resource & data management, security, QoS, policy, communication, adaptation, etc.
- Development of Globus Toolkit
  - Core services for grid-enabled tools & apps
- Construction of production grids & testbeds
  - Multiple deployments to distributed organizations for production & prototyping
- Application experiments
  - Distributed applications, tele-immersion, etc.

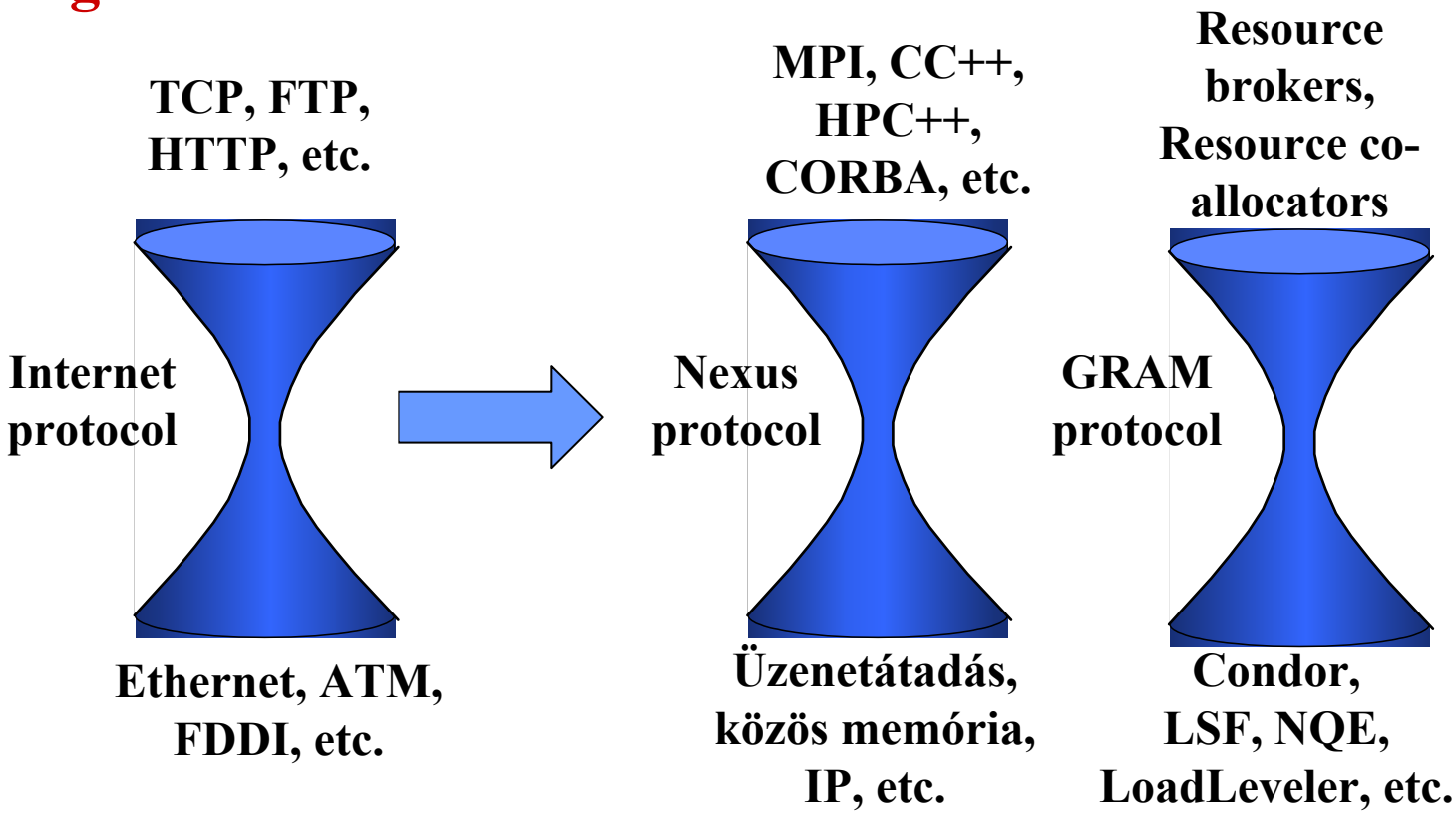
Globus Approach: Hourglass

- 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



Globus Approach: Hourglass

High-level services



Low-level tools

Globus Toolkit Grid Services

- Security (GSI)
- Resource management (GRAM)
- Information services (MDS)
- Remote file management (GASS)
- Communication (I/O, Nexus)
- Process monitoring (HBM)

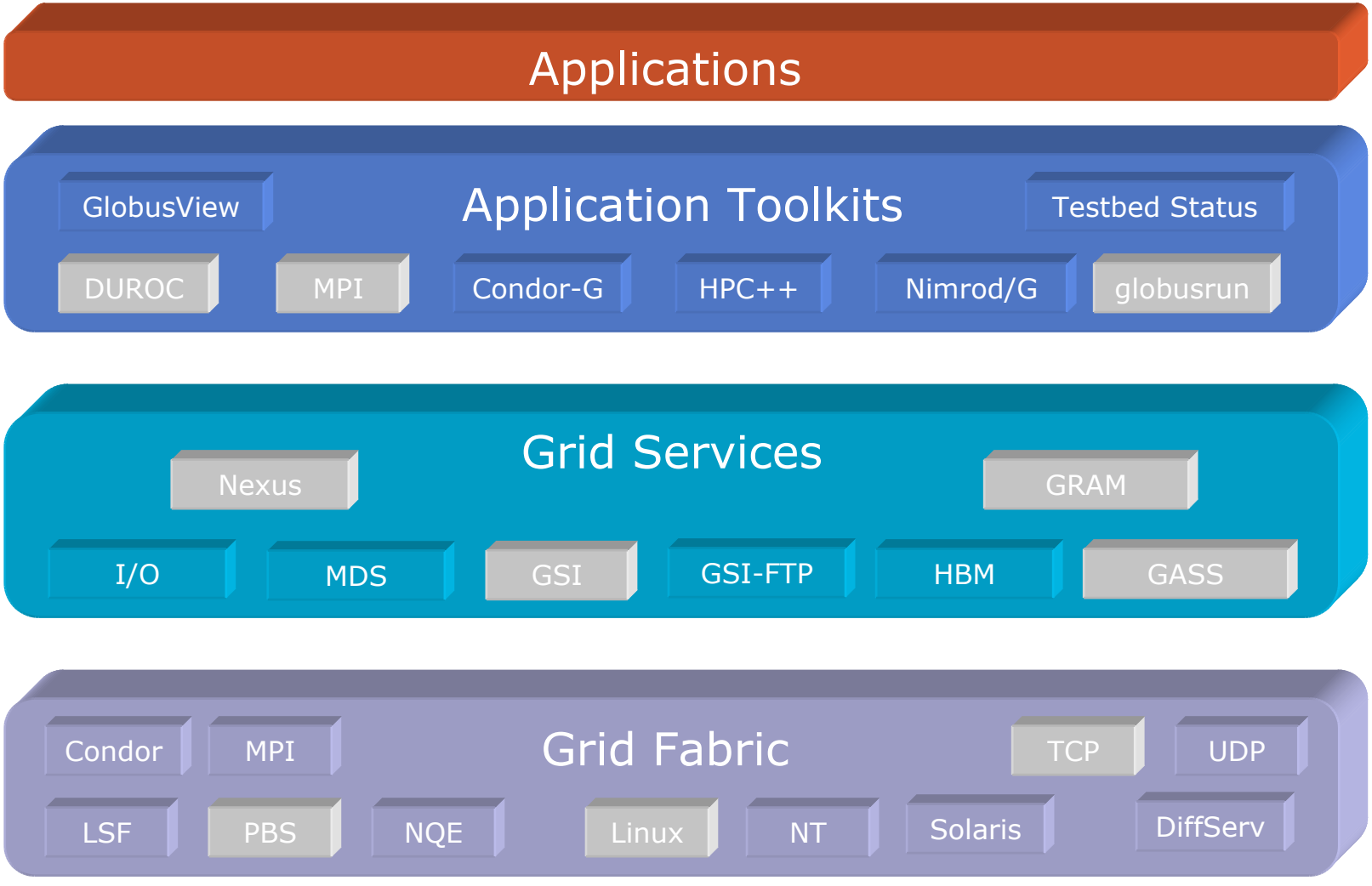
Other Globus Project Grid Services

- Coming Soon
  - Data transfer (GSI-FTP)
  - Replica Management

    http://www.globus.org/datagrid

- Experimental Prototypes
  - Advanced Reservations & QoS (GARA)
  - Distributed Events & Logging

Layered Architecture

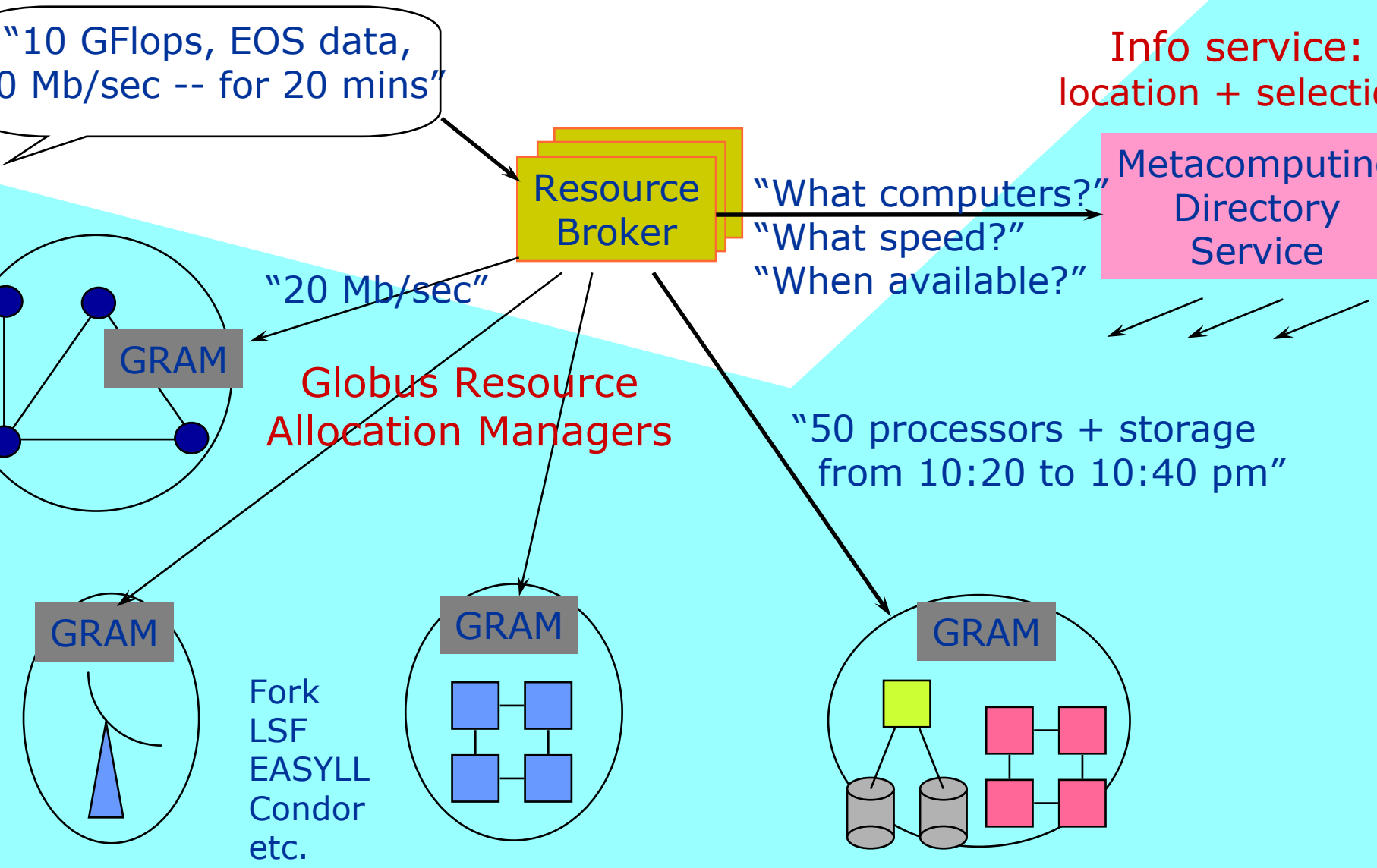


The Need for Information System

- System information is critical to operation of the grid and construction of applications
  - How does an application determine what resources are available?
  - What is the "state" of the computational grid?
  - How can we optimize an application based on configuration of the underlying system?
- We need a general information infrastructure to answer these questions

the globus project
www.globus.org

Using Information for Resource Brokering



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 Service

- Provide access to static and dynamic information regarding system components
- Requirements and characteristics
  - Uniform, flexible access to information
  - Scalable, efficient access to dynamic data
  - Access to multiple information sources
  - Decentralized maintenance

The Globus Toolkit Metacomputing Directory Service

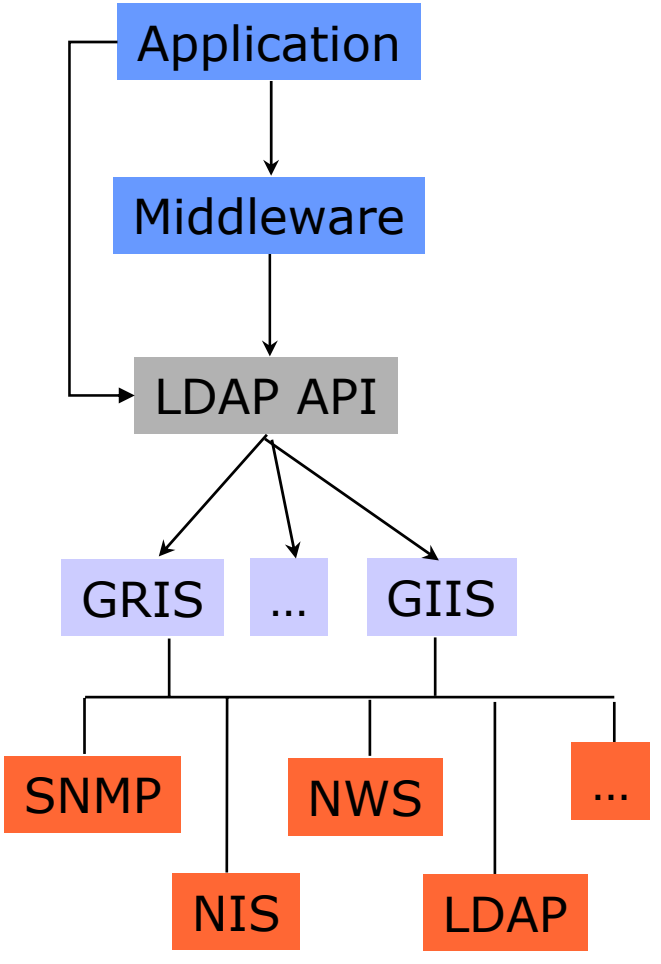
- Store information in distributed directories
  - Directory stored in collection of LDAP servers
  - Each server optimized for particular function
- Directory can be updated by
  - Information providers and tools
  - Applications (i.e., users)
  - Backend tools which generate info on demand
- Information dynamically available to
  - Tools
  - Applications

Directory Service Functions

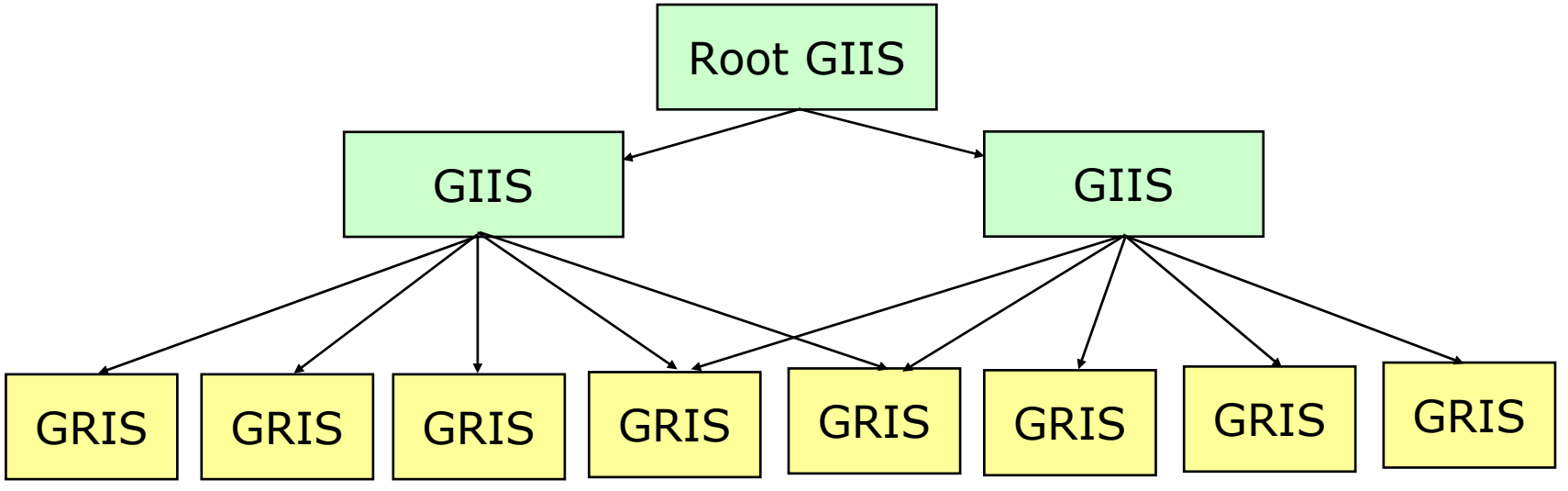
- White Pages
  - Look up the IP number, amount of memory, etc., associated with a particular machine
- Yellow Pages
  - Find all the computers of a particular class or with a particular property
- Temporary inconsistencies are often considered okay
  - In a distributed system, you often do not know the state of a resource until you actually use it
  - Information is often used as “hints”

MDS Approach

- Based on LDAP
  - Lightweight Directory Access Protocol v3 (LDAPv3)
  - Standard data model
  - Standard query protocol
- Globus specific schema
  - Host-centric representation
- Globus specific tools
  - GIIS (Grid Index Info Server)
  - GRIS (Grid Resource Info Server)
  - Data discovery, publication,…



Hierarchical Grid Resource Discovery System

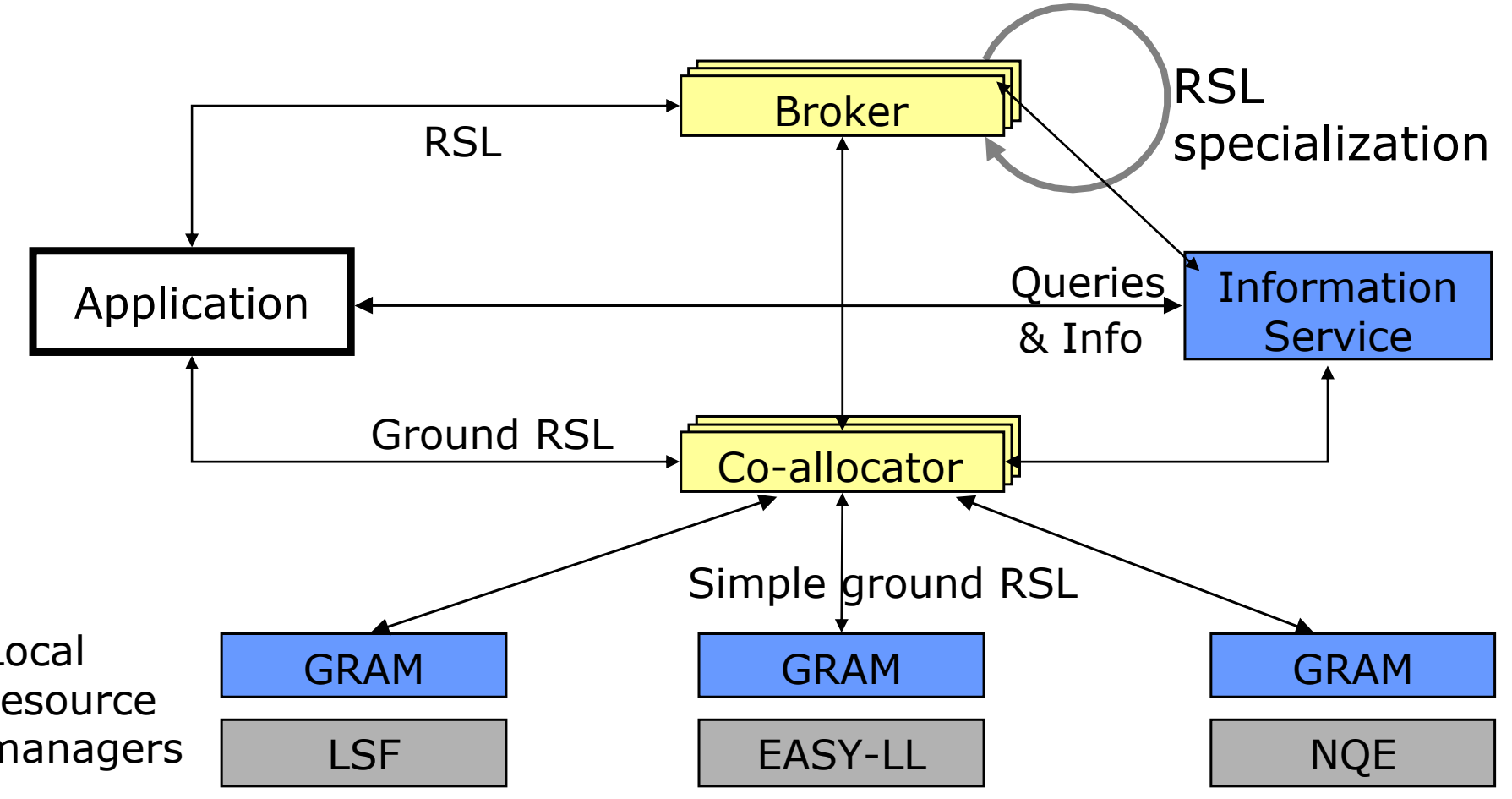


1. Once the client found the root service, it submits its query to it
2. The root GIIS returns a set of possible candidates, using the static attributes
3. The client uses this set of information to narrow the search to a child GIIS
4. The client uses the new set of information to narrow the search to a GRIS and to obtain dynamic information of the selected resource

Components of Resource Management

- Resource Specification Language (RSL) is used to communicate requirements
- The Globus Resource Allocation Manager (GRAM) API allows programs to be started on remote resources
- A layered architecture allows application-specific resource brokers and co-allocators to be defined in terms of GRAM services

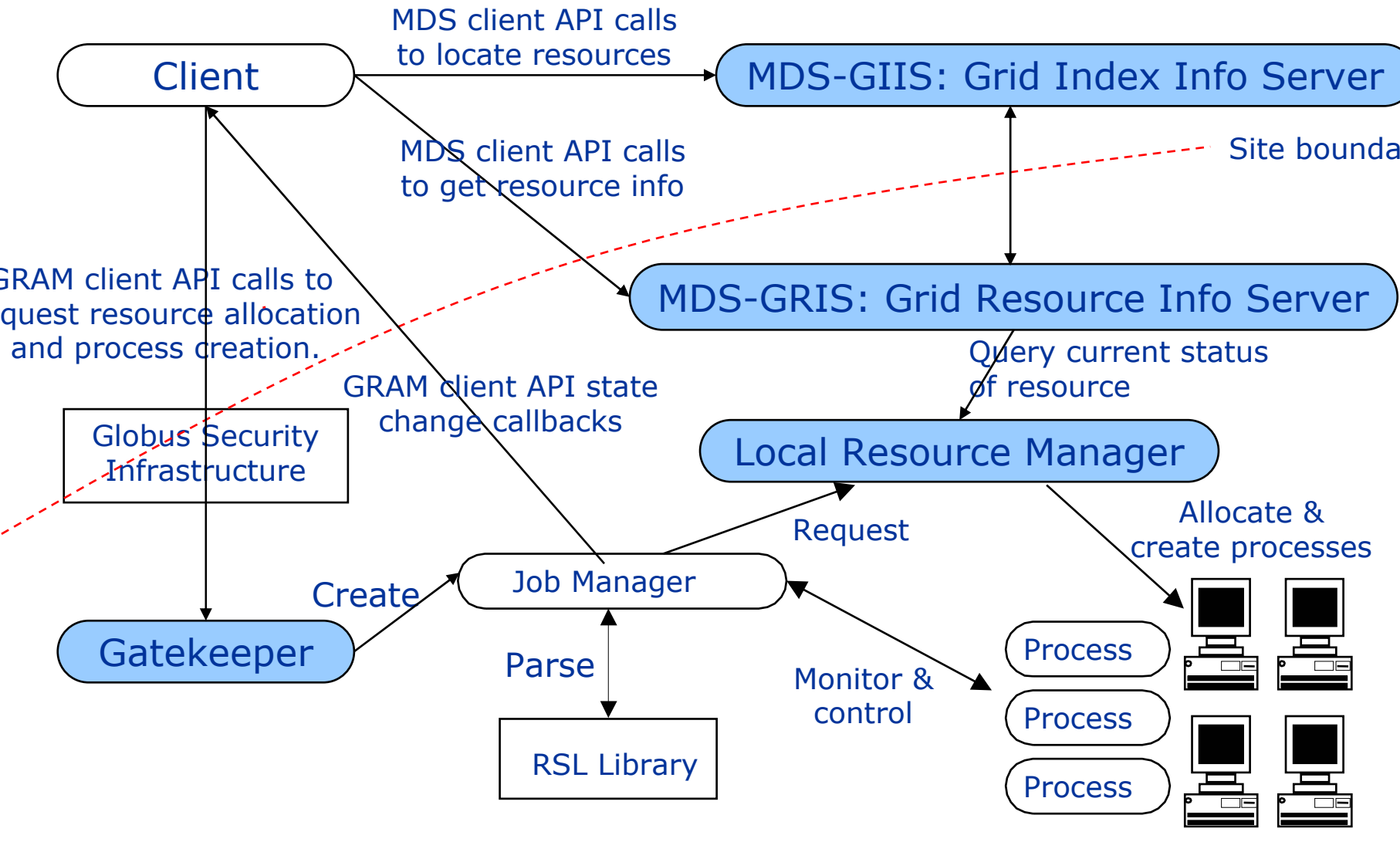
Resource Management Architecture



Globus Resource Allocation Manager (GRAM)

- Processing RSL specifications representing resource requests, by
  - Either creating the process(es) that satisfies the request
  - Or denying that request
- Periodically updating the MDS with info about the current status of the resources it manages
- Enabling remote job monitoring and management

GRAM Components



Globus Ports

GRAM: 2119
GRIS: 2135
GIIS port (system administrator can set up)
GASS port (user can set up)

Gatekeeper:

When a request arrives on a Globus port, the Gatekeeper checks the security aspects. If they are fine, the Gatekeeper initiates the requested service on the port. These services are:

- MDS
- GRAM
- GASS

Resource Specification Language

- Common notation for exchange of information between components
  - Syntax similar to MDS/LDAP filters
- RSL provides two types of information:
  - Resource requirements: Machine type, number of nodes, memory, etc.
  - Job configuration: Directory, executable, args, environment
- API provided for manipulating RSL

RSL Syntax

- Elementary form: parenthesis clauses
  - (attribute op value [value …])
- Operators Supported:
  - <, <=, =, >=, > , !=
- Some supported attributes:
  - executable, arguments, environment, stdin, stdout, stderr, resourceManagerContact, resourceManagerName
- Unknown attributes are passed through
  - May be handled by subsequent tools

Constraints: "&"

- For example:

  "Create 5-10 instances of myprog, each on a machine with at least 64 MB memory that is available to me for 4 hours"

  & (count>=5) (count<=10)
  (max_time=240) (memory>=64)
  (executable=myprog)

Disjunction: "|"

- For example:
- Create 5 instances of myprog on a machine that has at least 64MB of memory, or 10 instances on a machine with at least 32MB of memory

```
& (executable=myprog)
   ( | (&(count=5)(memory>=64))
       (&(count=10)(memory>=32)))
```

Multirequest: "+"

- A multirequest allows us to specify multiple resource needs, for example

```
+ (& (count=5)(memory>=64)
        (executable=p1))
   (&(network=atm) (executable=p2))
```

  - Execute 5 instances of p1 on a machine with at least 64M of memory
  - Execute p2 on a machine with an ATM connection
- Multirequests are central to co-allocation

Co-allocation

- Simultaneous allocation of a resource set
  - Handled via optimistic co-allocation based on free nodes or queue prediction
  - In the future, advance reservations will also be supported
- globusrun and globus-job-* will co-allocate specific multi-requests
  - Uses a Globus component called the Dynamically Updated Request Online Co-allocator (DUROC)

A Co-allocation Multirequest

```
+( & (resourceManagerContact=
     "flash.isi.edu:754:/C=US/…/CN=flash.isi.edu-fork")
     (count=1)
     (label="subjob A")
     (executable=my_app1)
   )
   ( & (resourceManagerContact=
     "sp139.sdsc.edu:8711:/C=US/…/CN=sp097.sdsc.edu-lsf")
     (count=2)
     (label="subjob B")
     (executable=my_app2)
   )
```

Different counts

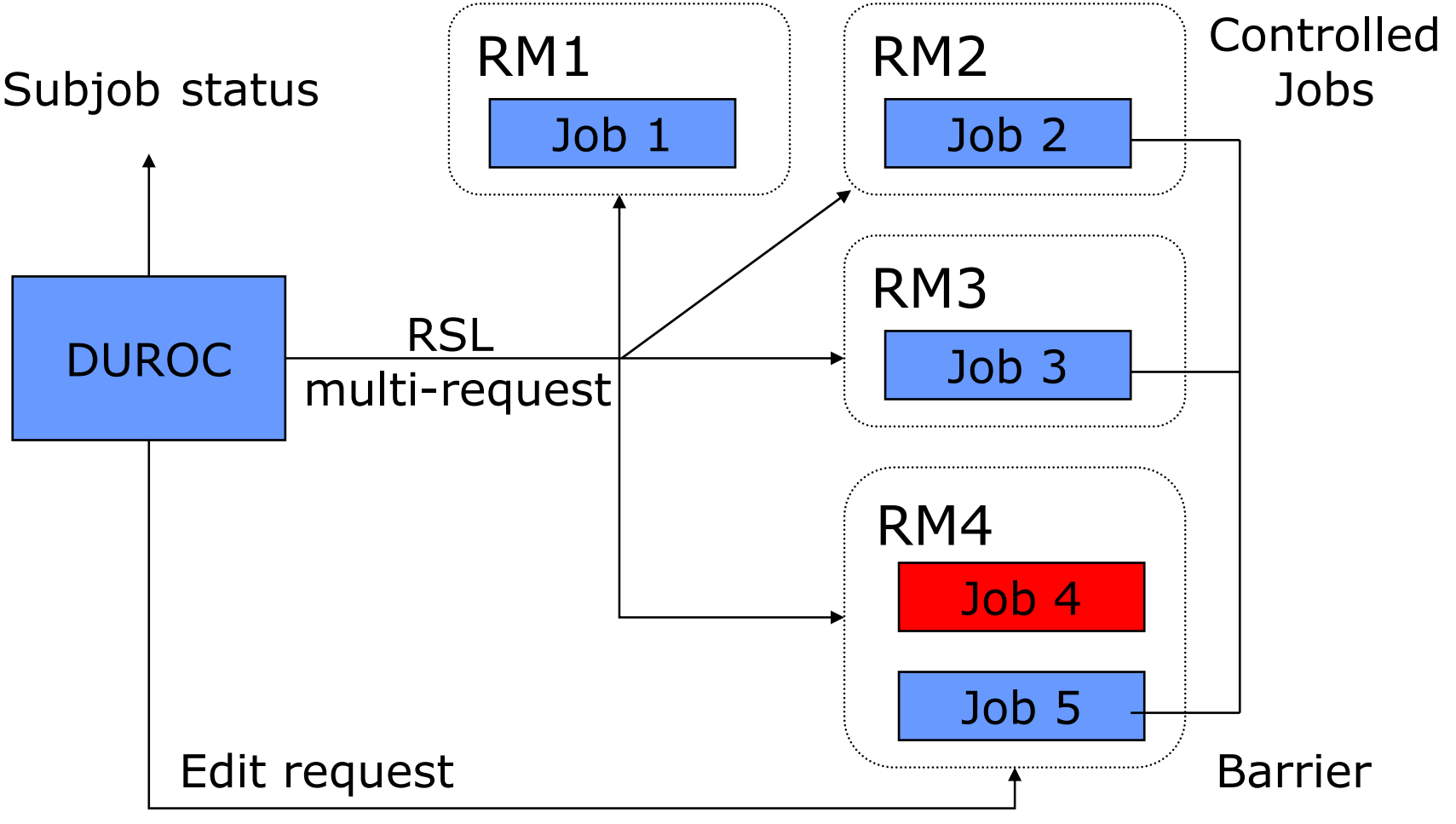
Different resource managers

Different executables

DUROC Functions

- Submit a multi-request
- Edit a pending request
  - Add new nodes, edit out failed nodes
- Commit to configuration
  - Delay to last possible minute
  - Barrier synchronization
- Initialize computation
  - Bootstrap library
- Monitor and control collection

DUROC Architecture



Global Access to Secondary Storage (GASS)

GASS enables access to remote files and executables

(a) GASS file access API

- Replace open/close with globus_gass_open/close; read/write calls can then proceed directly

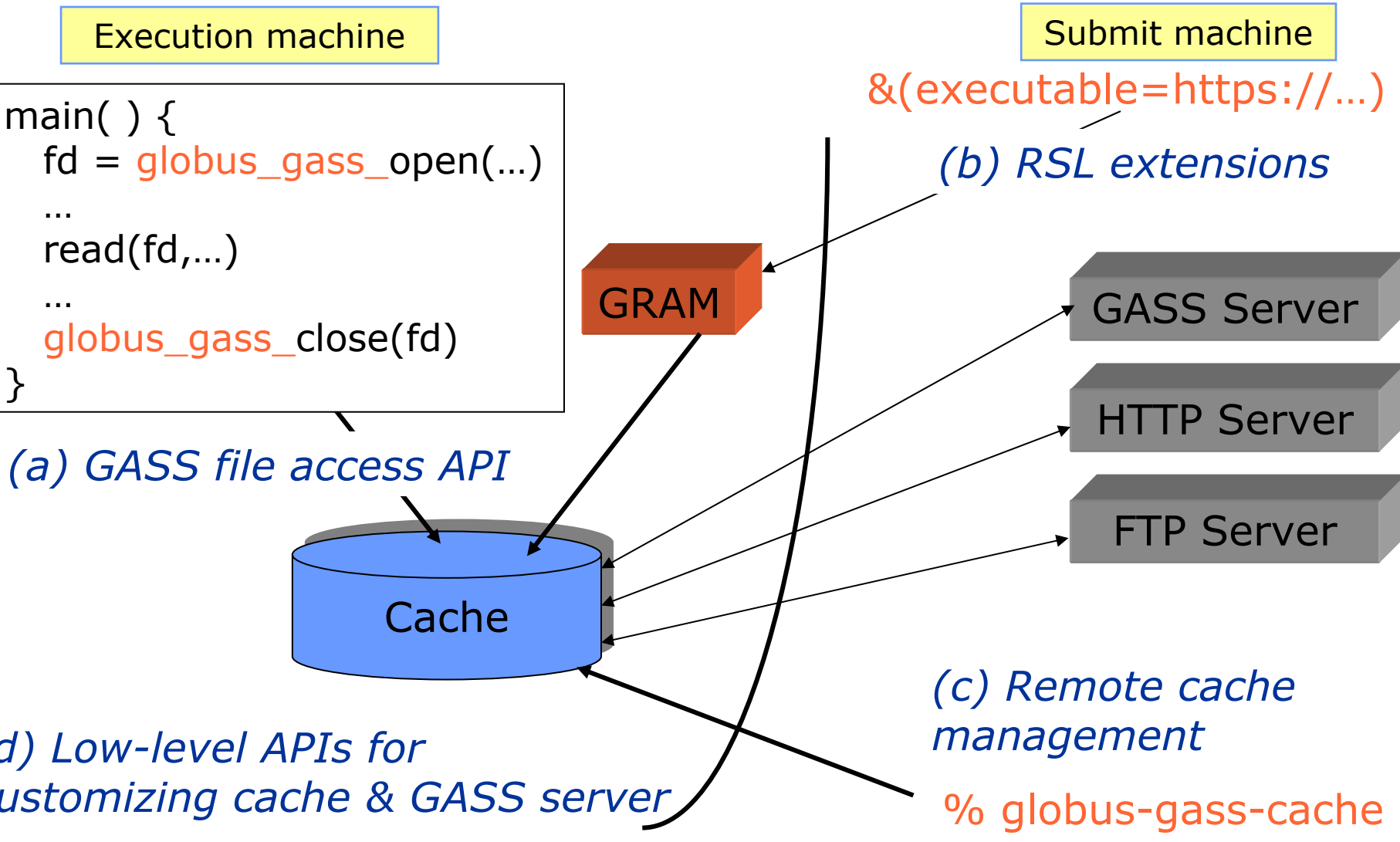
(b) RSL extensions

- URLs used to name executables, stdout, stderr

(c) Remote cache management utility

(d) Low-level APIs for specialized behaviors

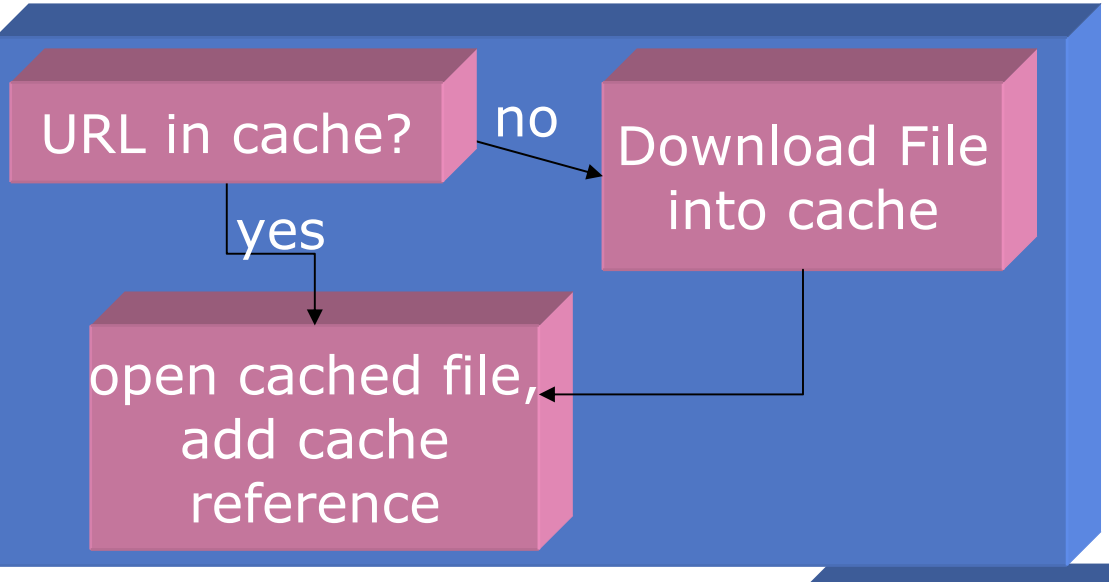
GASS Architecture



GASS/RSL Example

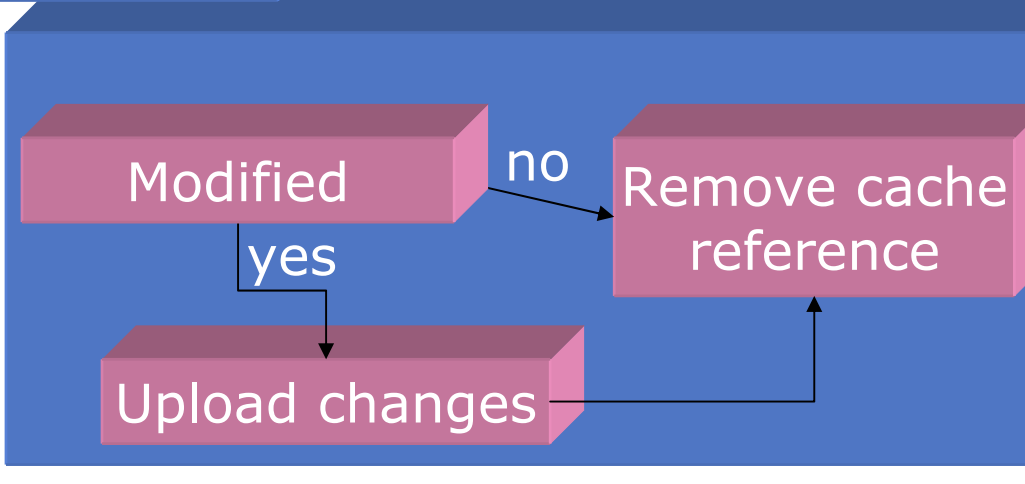
```
&(executable=https://quad:1234/~/myexe)
  (stdin=https://quad:1234/~/myin)
  (stdout=/home/bester/output)
  (stderr=https://quad:1234/dev/stdout)
```

globus_gass_open()/close()



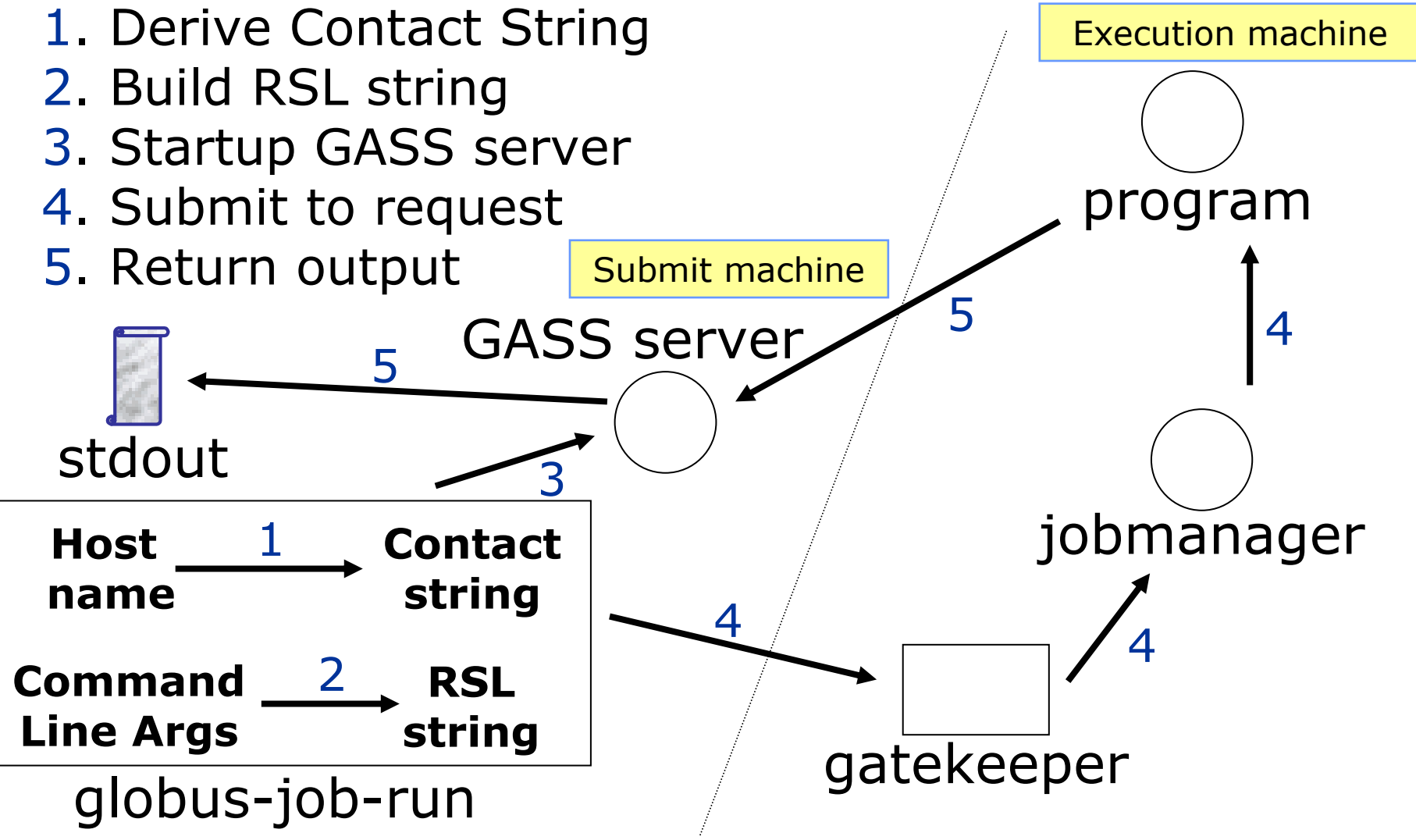
globus_gass_open()

globus_gass_close(

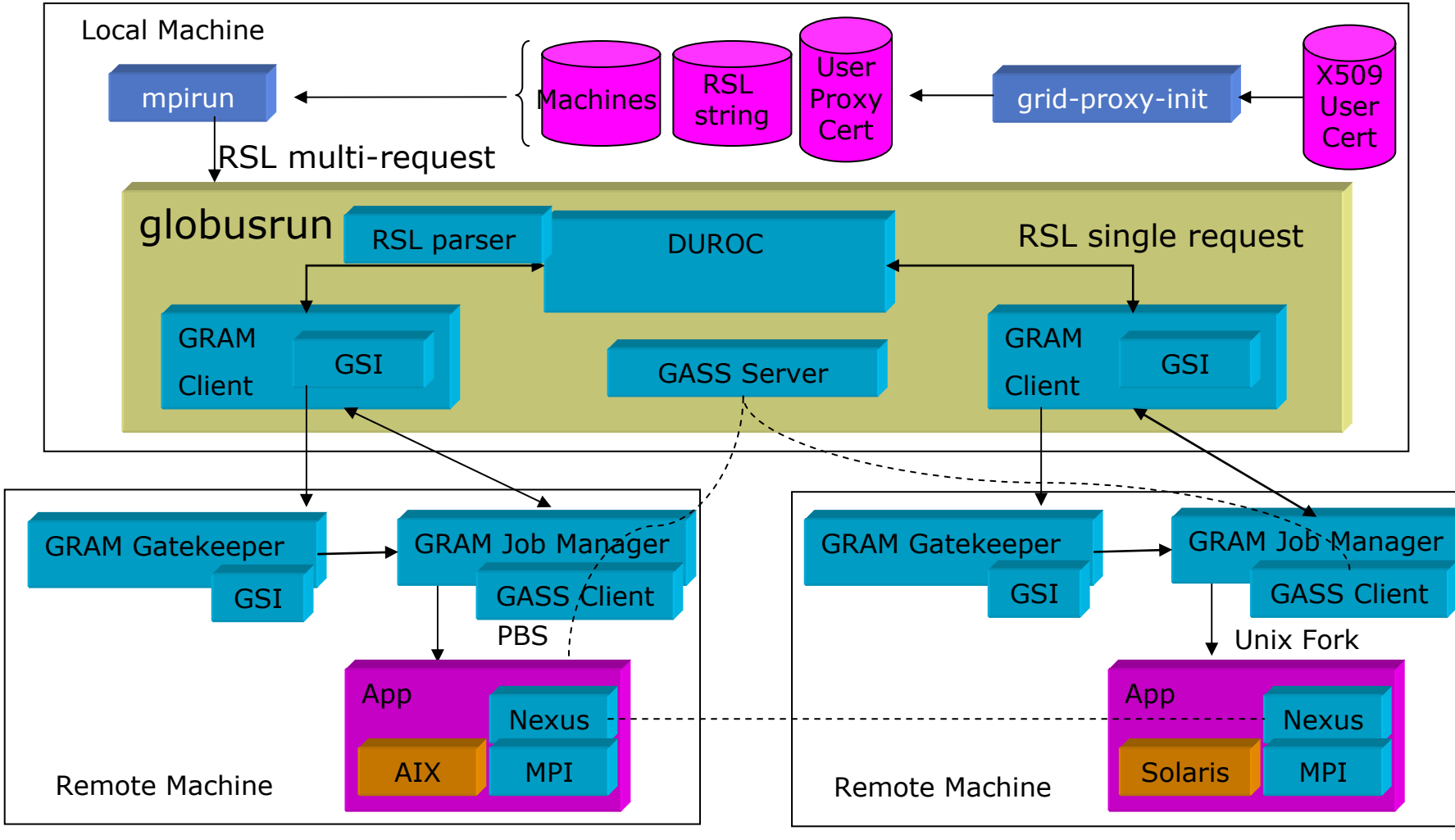


GRAM & GASS: Putting It Together

1. Derive Contact String
2. Build RSL string
3. Startup GASS server
4. Submit to request
5. Return output



Globus Components In Action



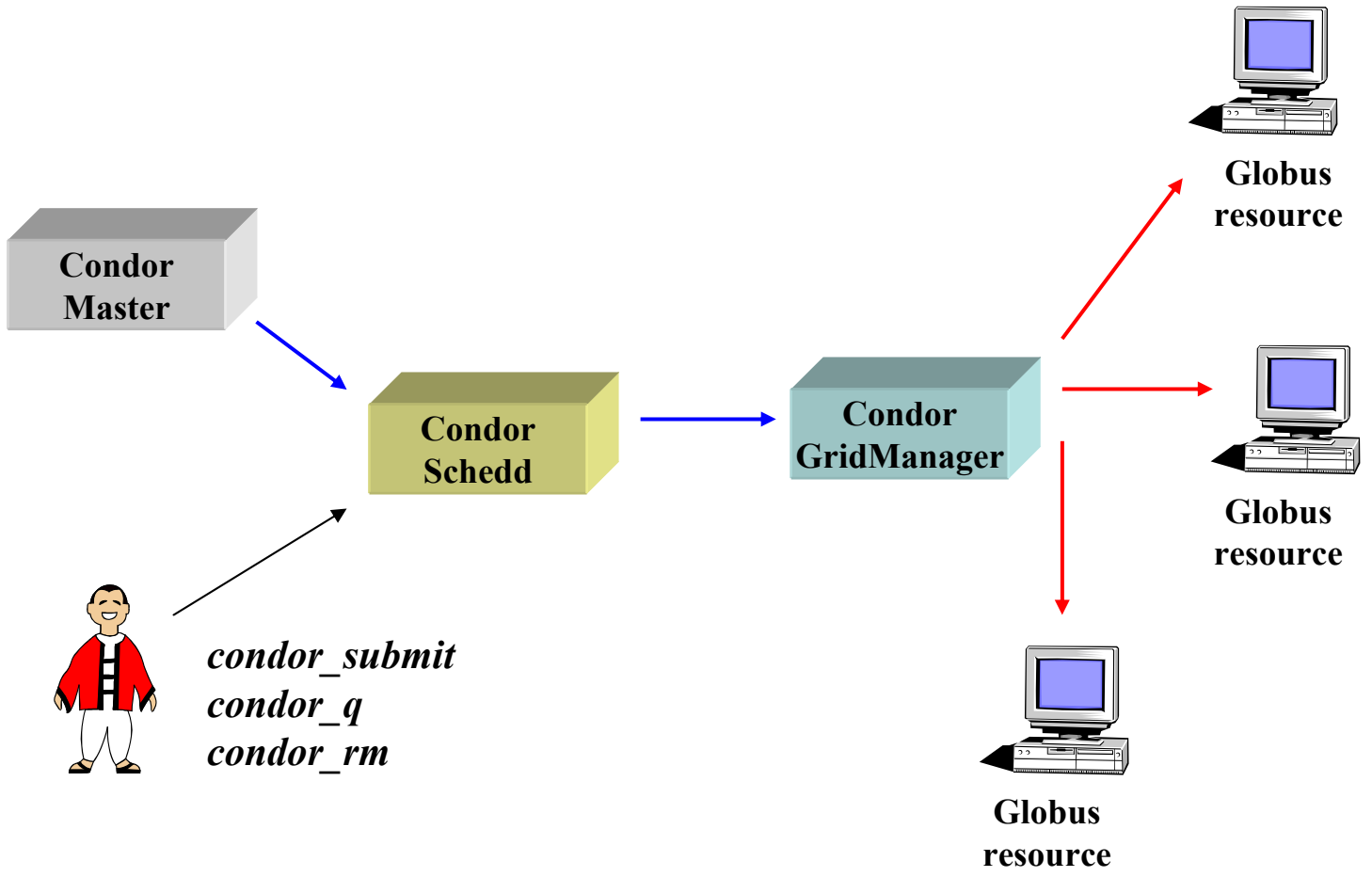
What is Condor-G?

- Condor-G is a Personal-Condor enhanced with Globus services
- It knows how to speak to Globus resources via GRAM
- It can be used to submit jobs to remote Globus resources
- It makes Condor keep track of their progress

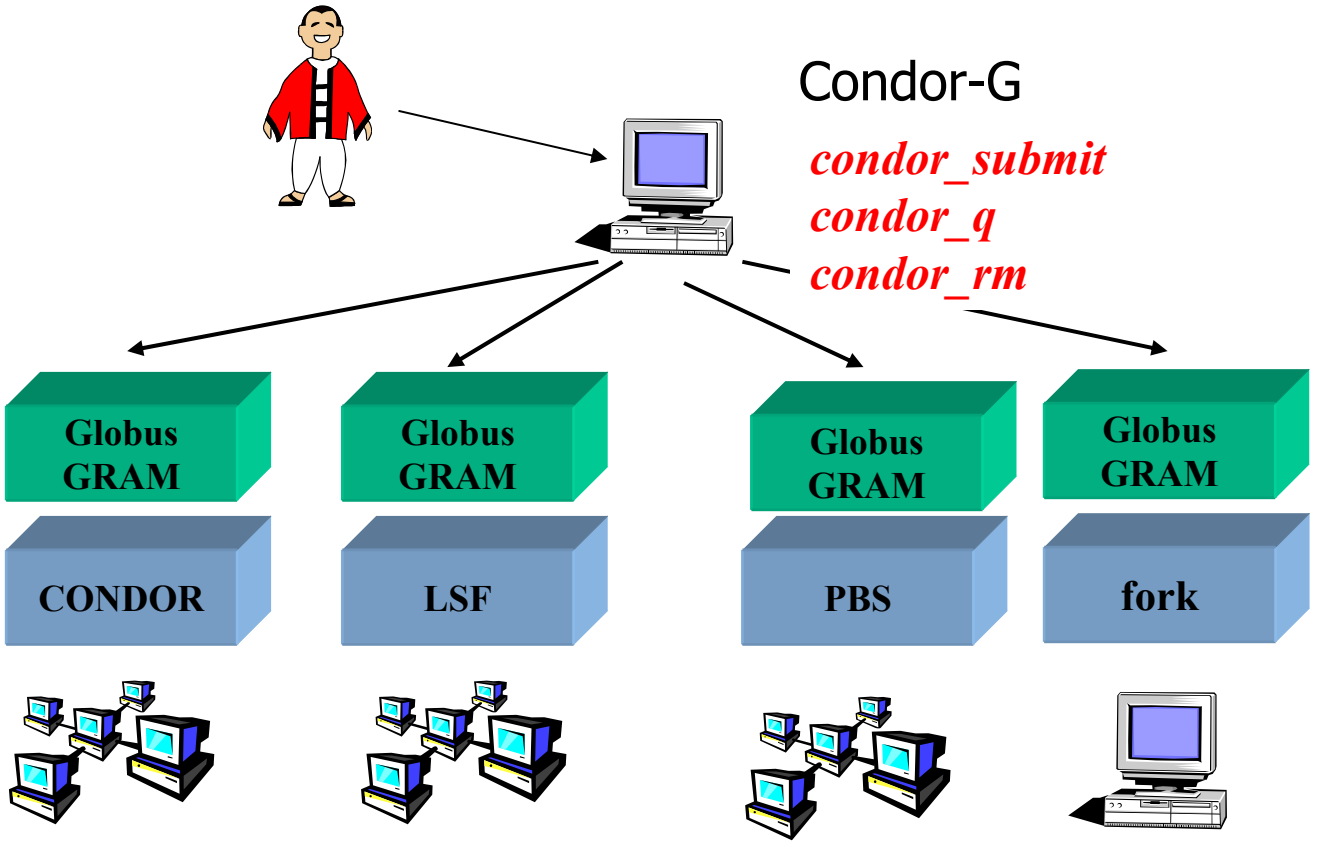
Condor-G: Condor for the Grid

- Condor is a high-throughput scheduler
- Condor-G uses Globus Toolkit libraries for:
  - Security (GSI)
  - Managing remote jobs on Grid (GRAM)
  - File staging & remote I/O (GSI-FTP)
- Grid job management interface & scheduling
  - Robust replacement for Globus Toolkit programs
    - To implement a reliable, crash-proof, checkpointable job submission service
  - Supports single or high-throughput apps on Grid
    - Personal job manager which can exploit Grid resources

The Use of Condor-G



Condor-G as user job submission service



the globus project
www.globus.org

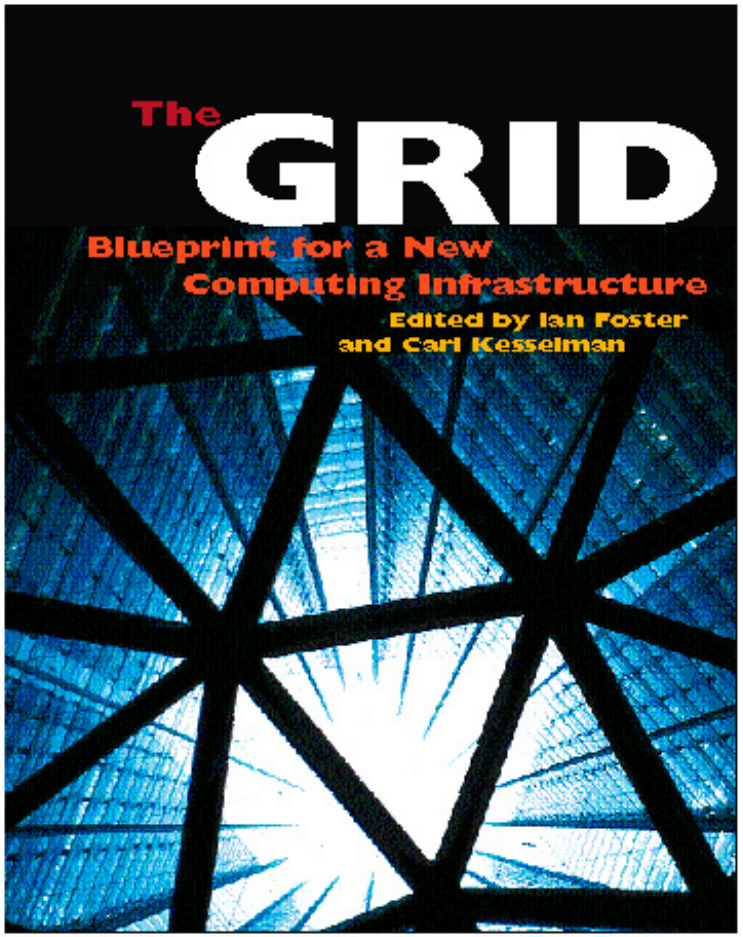
The Grid:
Blueprint for a New Computing Infrastructure

I. Foster, C. Kesselman (Eds), Morgan Kaufmann, 1999

- Available July 1998;

  ISBN 1-55860-475-8
- 22 chapters by expert authors including Andrew Chien, Jack Dongarra, Tom DeFanti, Andrew Grimshaw, Roch Guerin, Ken Kennedy, Paul Messina, Cliff Neuman, Jon Postel, Larry Smarr, Rick Stevens, and many others

"A source book for the history of the future" -- Vint Cerf



http://www.mkp.com/grids