Grid Programming: Concepts and Challenges

Michael Rokitka
SUNY@Buffalo
CSE510B
10/2007

Issues Due to Heterogeneous Environment

- Hardware level
  - Different architectures, chipsets, execution speeds
- Software level
  - Different operating systems (and versions), different compilers, heterogeneous software environments (libraries, etc.)
- Administrative level
  - Differing and incompatible administrative policies between various grid resources

Key Issues for Grid Applications

- Resource reservation
- Security
- Accounting (Logging & Audit Trail)
- Communication (IPC)
- Rapidly changing runtime environment
- Grid is still in many ways in its infancy
  - There are many APIs of varying size and complexity
  - APIs are subject to (possibly rapid) change, just as the Grid is
  - Programmers often add a further layer of abstraction on top of these APIs to simplify dealing with them

From an Application Programmer's Perspective

- Who is the typical application programmer?
  - Domain scientist trying to solve domain specific problems
  - NOT a Grid expert, or software expert
  - Would prefer to just work on solving their problem, not having to spend time worrying about low-level details.
- Great need for standardized and widely adopted programming interfaces

Grid Service & Middleware APIs

- SOAP based services
  - SDL: Service Description Language
- Protocol based services
  - Accessed via client side C API, well defined protocol
- Web Service based services
  - WSDL & WSRF
- Still others
  - Complex APIs, easier to use tools provided

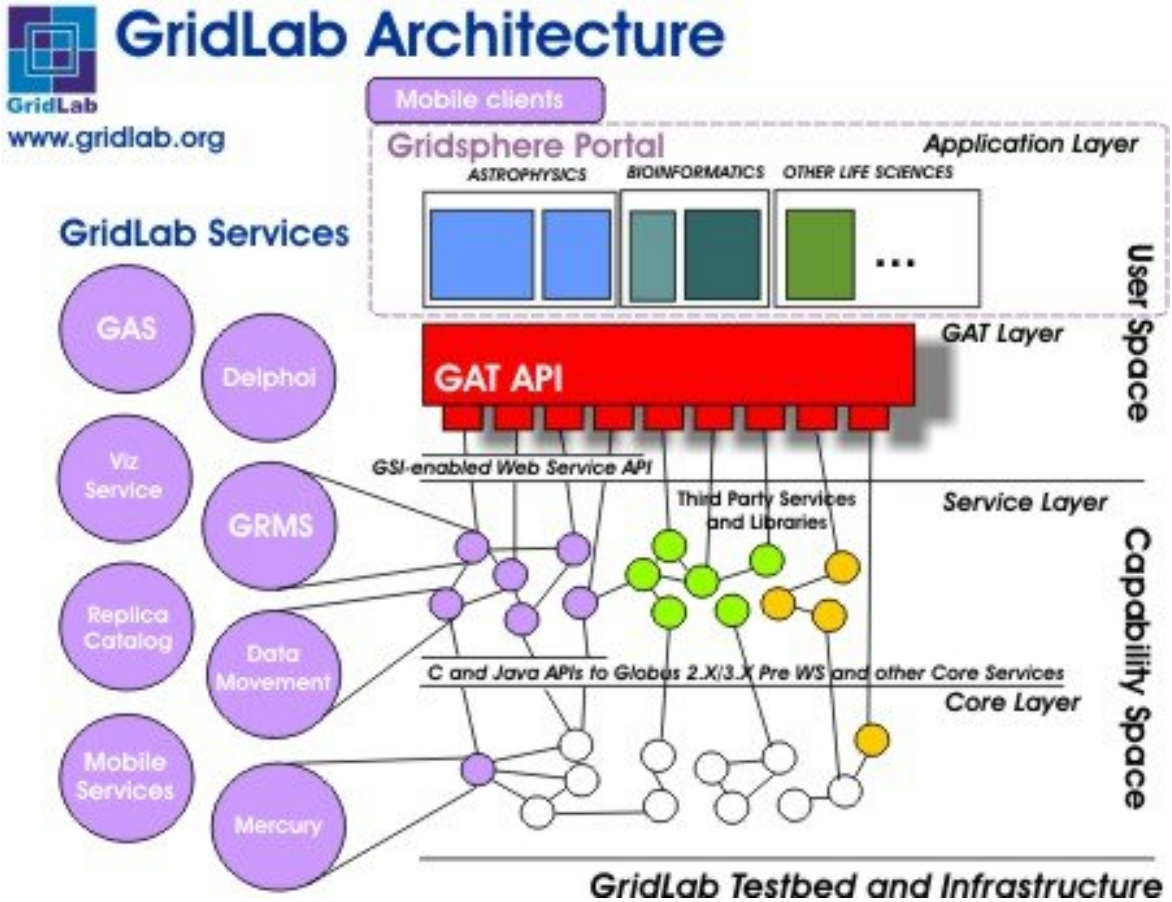
The Need for Uniformity

- Currently there are too many APIs, many of which are complex, for dealing with common services
- The learning curve for the various frameworks and APIs needed is high
- Applications need to be updated continuously to keep up with API & framework changes
- A uniform approach to Grid APIs, or even a single Grid API is needed to deal with these issues

GAT – The Grid Application Toolkit

- What is GAT?
  - GAT is a set of coordinated, generic and flexible APIs for accessing Grid services from e.g. generic application codes, portals, data managements systems, together with working implementations provided by the tools developed in the Grid Lab project
  - GAT is designed in a modular plug-and-play manner, such that tools developed anywhere can be plugged into GAT
  - GAT itself is a library which presents to the application programmer a uniform interface to Grid technologies. The actual Grid technologies which implement the GAT API functionalities are plugged into GAT by means of a plugin architecture
    - Allows the application programmer to worry about the application and the Grid experts to worry about the plugins which talk to their services

GAT API & the GridLab Architecture



How Does GAT Work?

- The client application makes GAT API calls for operations which may be Grid-related
- The client application links against the GAT Engine
- The client application runs irrespective of actual underlying infrastructure deployment
- The GAT engine loads adaptors which are valid in the environment extant when the application starts
- The GAT adaptors try to do Grid operations on request, on failure another adaptor provided function may be called

SAGA - Simple API for Grid Apps

- What is SAGA?
  - An object-oriented API with a number of functional packages for fundamental programming capabilities
  - Provides a high-level interface to different middleware
  - Language-independent, with multiple language possibility of multiple language bindings
    - Interface is specified in SIDL (Scientific Interface Description Language)

SAGA Functional Packages

- Job Management
  - Provides job submission to grid resources and subsequent management, either in batch or interactive mode, using a state model consistent with that of OGSA-BES.
- Name Spaces
  - Provides methods for managing hierarchical name spaces
- File Management
  - In conjunction with the Name Space Package, provides methods for operating on the content of files, e.g., read, write, seek
- Replica Management
  - Provides logical files and replicas, along with search based on logical file metadata
- Streams
  - Provides a persistent byte stream abstraction for communication objects
- Remote Procedure Call
  - Provides a remote procedure call style of interaction consistent with the OGF GridRPC standard

SAGA “Look & Feel” Packages

- Task Model
  - Provides management of any asynchronous tasks (local or remote)
- Monitoring Model
  - Provides a callback mechanism whereby an application can be notified of object state changes
- Sessions
  - Allows sets of SAGA objects to be isolated and managed independently
- Contexts
  - Provides a security information container attached to a session
- Base Object
  - Provides essential methods for all SAGA objects, e.g., unique ID
- Attribute Interface
  - Provides a common interface for getting/setting object attributes
- Error Handling
  - Provides uniform error handling for all SAGA objects

Grid Services: Information Services

- Information Services
  - Discovery and monitoring of Grid resources & services
- Globus MDS
  - MDS collects information across multiple, distributed resources on a grid via **aggregator services** that collect real-time (or fairly recent) state information from registered information sources into an **index**
  - Collections of information can be queried through various interfaces (browser, command line, and Web Services)
- GridLab's iGrid
  - Distributed architecture is based on two kind of Information Services, iServe and iStore (GSI-enabled web services)
  - iServe services supplies information about a specific resource
  - iStore services aggregate information coming from registered iServe

Grid Services: Job Submission & Management

- Job Manager
  - Enables the site or grid administrator to define and enforce procedures and policies for running jobs on a resource based on a wide range of properties such as computing system or type, user groups, priorities, run time, queue types and lengths, etc.
  - Provides end user with methods for submitting, monitoring, and controlling jobs
  - Eg. PBS, LSF, Globus GRAM, Torque
- Job Scheduler
  - Matches the job with the appropriate resources according to the requirements specified by the user (amount of memory, CPUs, disk space, running time, etc.)
  - Eg. Maui, PBS, LSF, Globus GRAM, Load Leveler

* Condor/Condor-G also provide this functionality

Grid Services: Advance Reservation

- AIST Grid Scheduling System (GRS)
  - Used for co-allocation of computing & network resources
    - Computing resource manager
    - Network resource manager
    - Grid resource scheduler that handles requests from users via the other two
- NAREGI GridVM
  - Provides a virtual execution environment and advanced registration of compute nodes
- Virtual Workspace (Keahey)
  - Execution environment in terms of the hardware and software components required
- Globus GRAM
  - Allows users to create and manage advance registration by leveraging the control provided by local resource manager

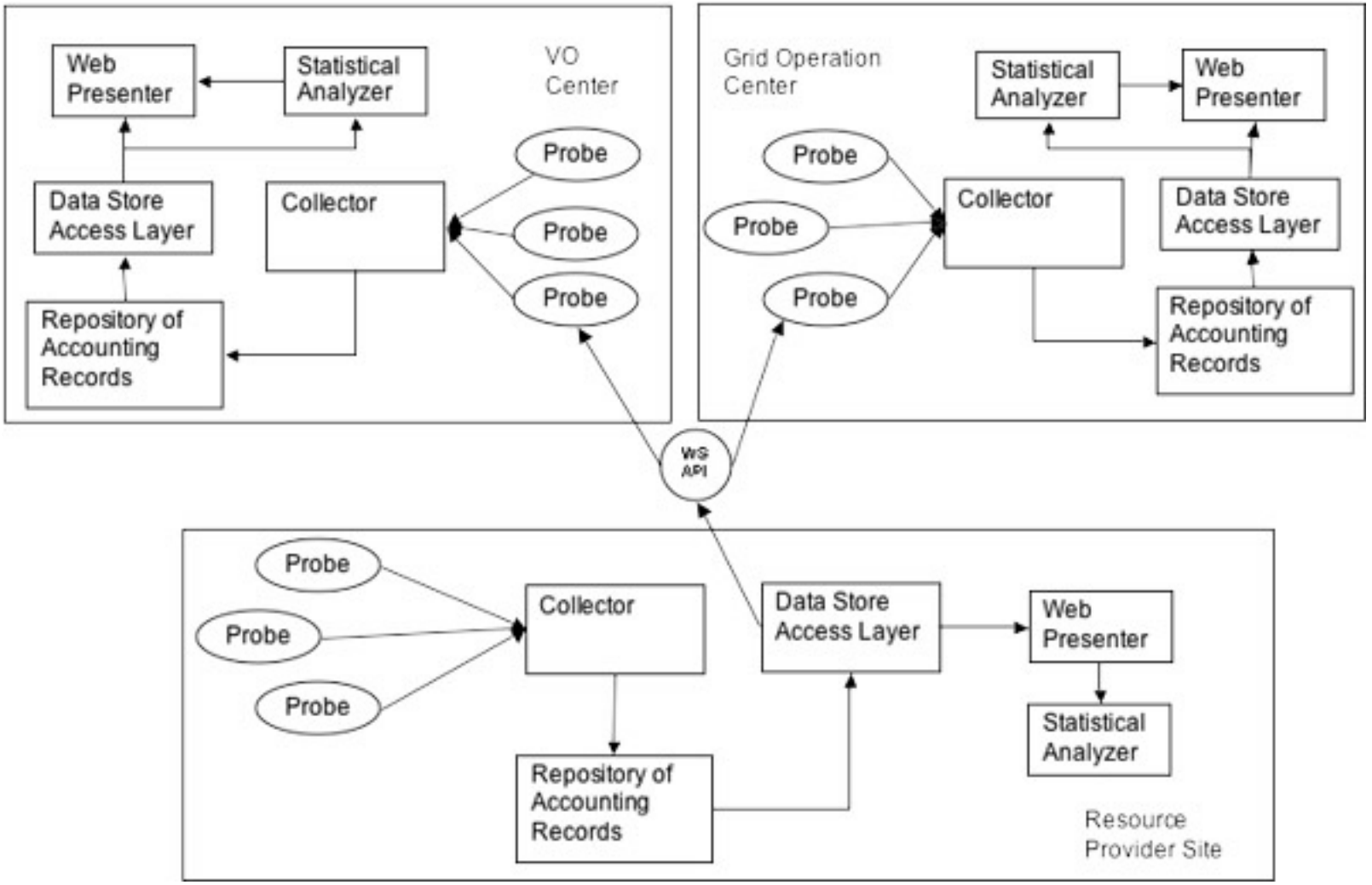
Data Access, Movement, and Storage

- Globus Services
  - GridFTP
    - Standard protocol based, requires open socket, non-recoverable
  - RFT
    - Web Service based, recoverable
- Condor Network Storage Technology (NEST)
  - Negotiates guaranteed storage allocations (contracts), in terms of "lots", between users and servers for specified periods of time
    - User/group access control
    - Storage automatically reclaimed at end of contract
    - Local limits and policies can be enforced by admin

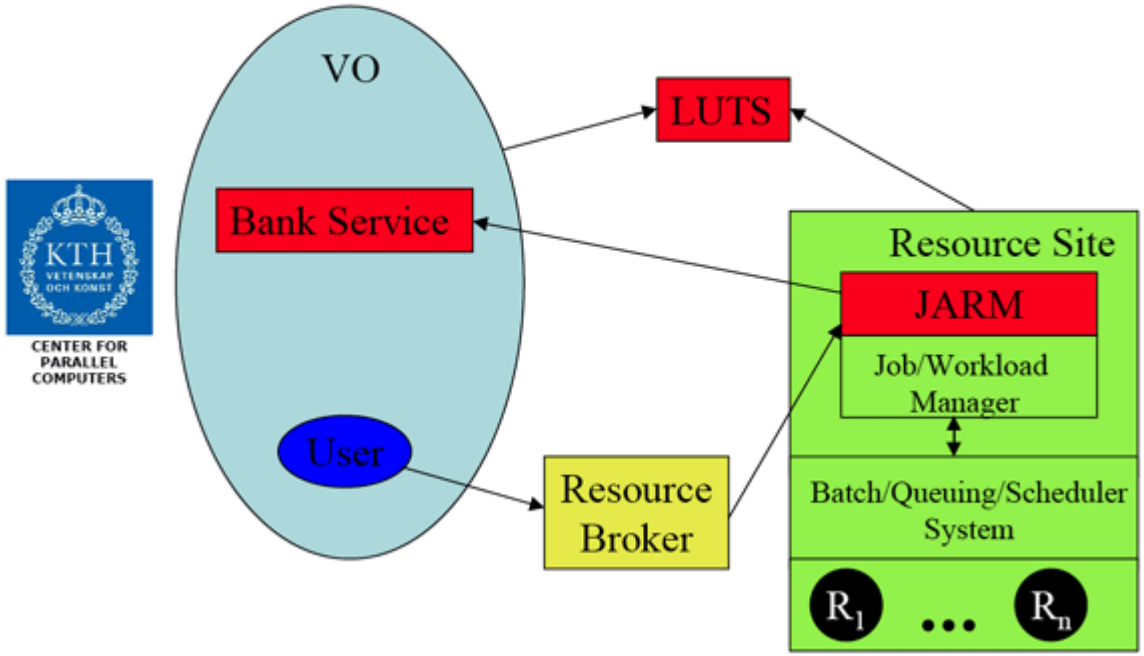
Accounting Packages

- Gratia
  - Large-scale operation on the Open Science Grid to collect accounting information
    - Data is collected via a standard process, running on each node, which generates daily usage logs containing information on the jobs that ran and how many resources they consumed
    - Probe is used to read generated files and convert them to usage records that the can then be sent to the collector
    - Data is consumed by a collector and stored in a reporting database
- SweGrid Accounting System (SGAS)
  - Java implementation of a resource allocation enforcement and tracking service based on the Web Service technologies
  - Includes logging and tracking in GGF Usage Record XML format and a remote and scriptable management interface

Gratia Architecture



SGAS Architecture



Workflow Processing

- *Workflow* is the operational aspect of a work procedure: how tasks are structured, who performs them, what their relative order is, how they are synchronized, how information flows to support the tasks and how tasks are being tracked
- Workflow problems can be modeled and analyzed using graph-based formalisms like Petri nets

- Condor Directed Acyclic Graph Manager (DAGman)
  - Allows you to specify the dependencies between Condor jobs
  - In case of failure at any step, DAGman will continue as far as possible and then create a "Rescue File" which holds the current state of the DAG job, which can later be used to restore the job to its previous state

Security

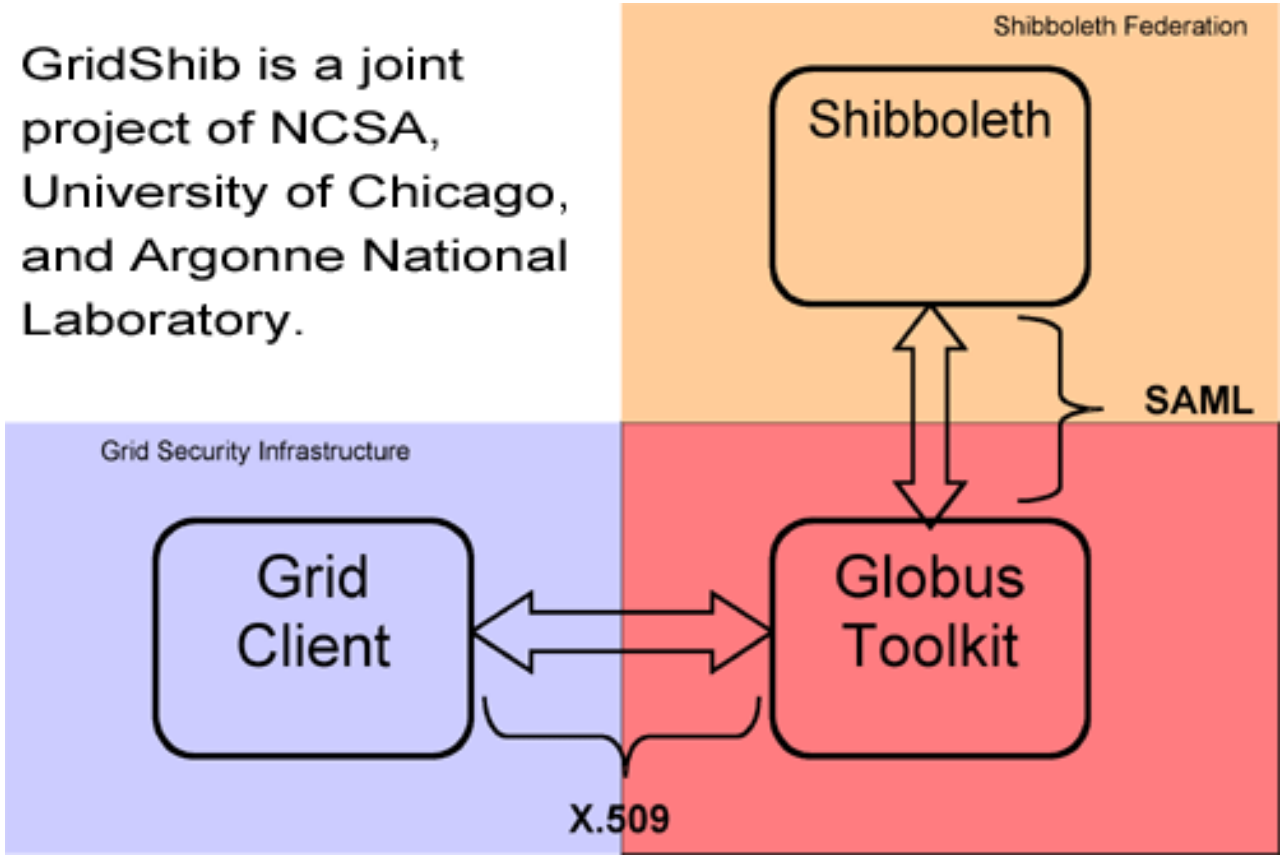
- GridShib
  - An NSF funded project of NCSA and the University of Chicago, to integrate the federated authorization infrastructure of Shibboleth with the Globus Toolkit
  - Identity-based authorization is provided via ACLs ("gridmaps") mapping to local identities
    (Unix logins) and a Community Authorization Service (CAS)
  - Shibboleth project offers a large base of campus use around the world via a standards-based and open source implementation and a standard vocabulary for describing user attributes.

GridShib Goals

- Integrate X.509 and SAML to provide enhanced Grid Security Infrastructure (GSI)
- Enable attribute sharing between virtual organizations and higher-educational institutions
- Develop and implement profiles to securely share attributes across administrative domains
- Investigate attribute-based access policy enforcement for grids
- Generalize attribute-based authorization policies in the Globus Toolkit runtime environment

GridShib Relationship

GridShib is a joint project of NCSA, University of Chicago, and Argonne National Laboratory.



References

- OGF SAGA one Pager
  - http://www.gridforum.org/UnderstandingGrids/documents/SAGA-OnePager%20jfe.pdf
- An introduction to SAGA
  - http://www.iceage-eu.org/issgc07/lectureMaterial/S25_saga_issgw07.pdf
- GridLab Grid Application Toolkit
  - http://www.gridlab.org/WorkPackages/wp-1/
- Grid Cookbook
  - http://www.sura.org/cookbook/gtcb/index.php?topic=7&mlevel=1&parent=0
- Shibboleth
  - http://shibboleth.internet2.edu/