Extensible Resource Broker for the Globus Toolkit

Enis Afgan
Purushotham Bangalore

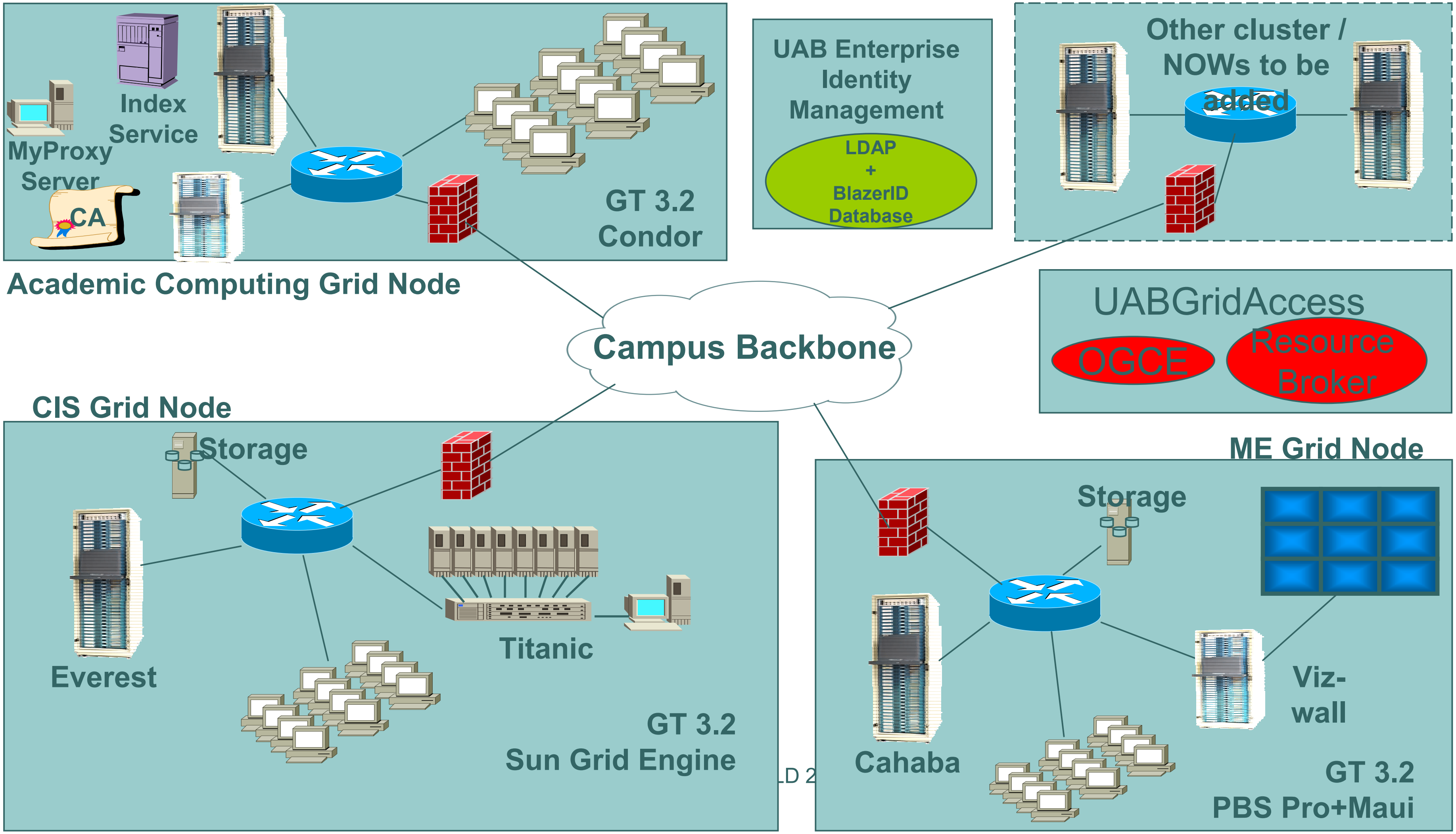
Collaborative Computing Lab (CCL)
Department of Computer and Information Sciences
University of Alabama at Birmingham (UAB)

GlobusWORLD 2005 – Boston, MA
February 10, 2005

Outline

- UABGrid
- Resource broker specifications
- Approach and architecture
- Scheduling algorithm
- Use cases
- Summary

UABGrid - in a picture



Academic Computing Grid Node

Index Service

MyProxy Server

CA

GT 3.2
Condor

UAB Enterprise Identity Management

LDAP + BlazerID Database

Other cluster / NOWs to be added

UABGridAccess

OGCE

Resource Broker

Campus Backbone

CIS Grid Node

Storage

Everest

Titanic

GT 3.2
Sun Grid Engine

ME Grid Node

Storage

Cahaba

Viz-wall

GT 3.2
PBS Pro+Maui

A resource broker (RB) specification

- In the environment with no control of local resources, it facilitates user’s resource selection and job submission
- Highly depends on Grid Information Services (GIS)
- Provides a new layer of abstraction thus creating true virtualization

Other approaches

- Nimrod G, AppLeS, Condor, MGrid
- Individual components oriented toward resource selection
- We’re building a complete solution that is simple to install and use orienting resource selection toward applications (*e.g.*, mpiBLAST)

Approach

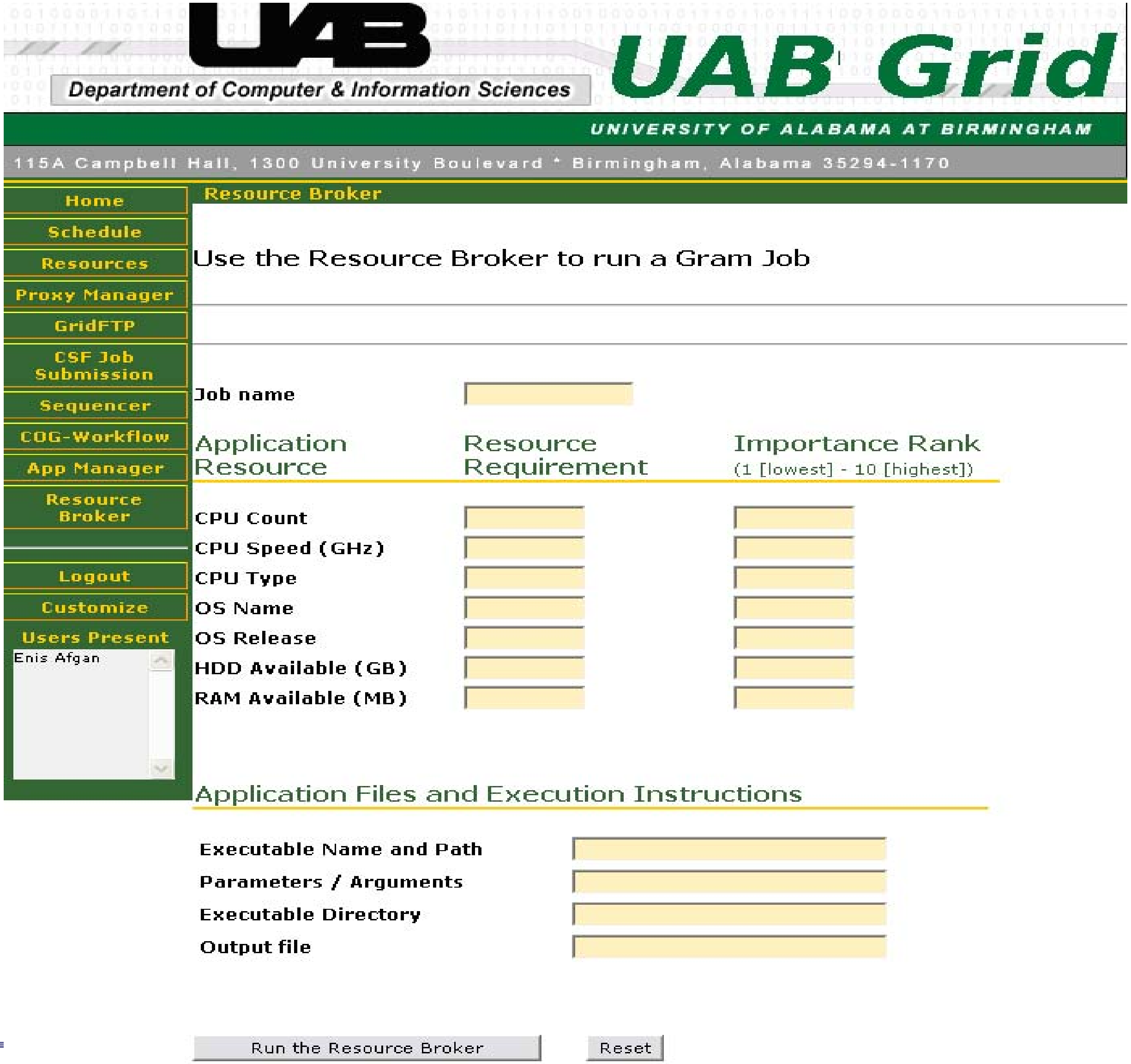
- Grid middleware technologies may be cumbersome to use
- Use web interface to receive user request
- Use application benchmarking
- Use load prediction

- Why such an approach?
  (information problem)

Adding RB to the existing infrastructure

- RB works on top of an existing infrastructure (requires MyProxy)
- Install using Ant
- With web interface eliminates any user-local software installations
- Single point of maintenance

User interface



UAB Grid

Department of Computer & Information Sciences

UNIVERSITY OF ALABAMA AT BIRMINGHAM

115A Campbell Hall, 1300 University Boulevard * Birmingham, Alabama 35294-1170

Resource Broker

Use the Resource Broker to run a Gram Job

Job name

Application Resource	Resource Requirement	Importance Rank (1 [lowest] - 10 [highest])
CPU Count		
CPU Speed (GHz)		
CPU Type		
OS Name		
OS Release		
HDD Available (GB)		
RAM Available (MB)		

Application Files and Execution Instructions

Executable Name and Path

Parameters / Arguments

Executable Directory

Output file

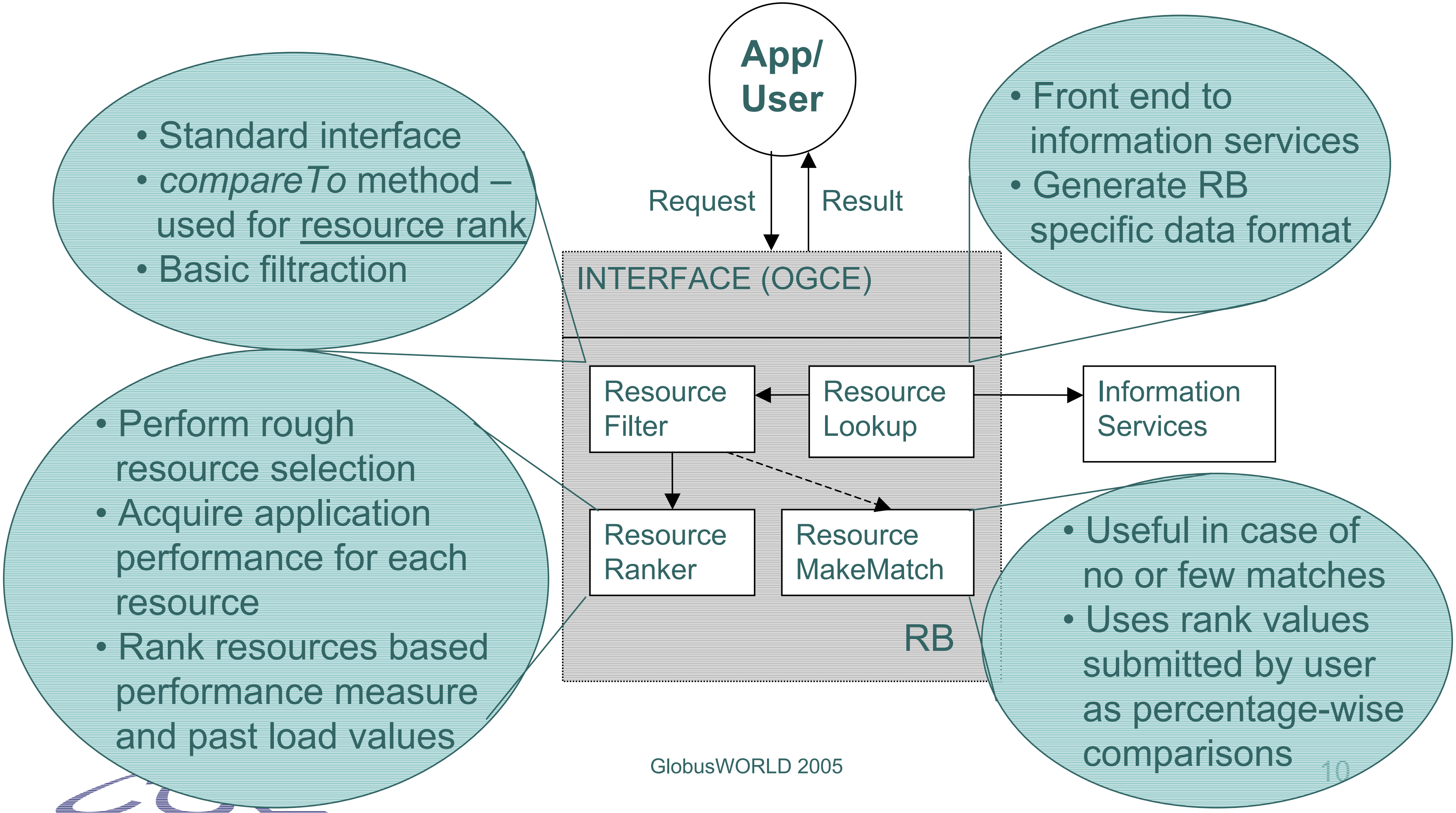
Users Present

Enis Afgan

Job monitoring

- Web based
- Users provides job name and current status reported
- Can be configured to notify the user when job completes
- Working on database keeping track of previous runs and parameters

Architecture



- Standard interface
- *compareTo* method – used for resource rank
- Basic filtraction

- Front end to information services
- Generate RB specific data format

- Perform rough resource selection
- Acquire application performance for each resource
- Rank resources based performance measure and past load values

- Useful in case of no or few matches
- Uses rank values submitted by user as percentage-wise comparisons

Application performance benchmarking

- Iteratively submit the job to all matching resources using sample problem set
- Use turnaround time as another variable when computing resource rank
- Currently: do it for each job submission with only current data
- Future: use stored application benchmarks to estimate time to completion related to individual resources & real data set

Fuzzy logic

- Fuzzy logic: rather than having only a true/false setting, an answer may have degree of truth between 0 and 1
- Excellent for control problems, since does not require noise-free inputs
- Rule-based system that assigns partial existence of an object to a set through the use of membership functions
- Small changes in parameters can result in large changes of the answer

Fuzzy logic scheduling

- As inputs to membership functions use:
  - application benchmark
  - static characteristics of a resource
  - user specified component relevancy
  - change in load variance
- Parameters in membership functions are statically typed
- Combine four results into the final resource rank

Why application benchmarking and fuzzy logic?

- Scheduling in the Grid = fuzzy benchmarking
  - Described lack of information
  - Uncertainty of received information
  - Heterogeneous machines performance reliability/dependability

MakeMatch component

- Still to be done (need run data)
- Accumulate application performance into database
- Create application specific profiles based on real runs
- Dynamically determine fuzzy logic parameters for scheduling

Use case

manual job submission

- Direct access to machine with user credentials – create proxy
- Query GIS and manually parse XML
- Decide which resource to choose
- Create RSL for job submission
- Submit the job
- Monitor the job from command line
- Transfer any output files back to local machine using GridFTP

Use case

using the Resource Broker for job submission

- Use web form to submit a request (include resource requirements, individual components rank values and the executable)
- Monitor job progress from a web page
- The user is notified when the job has been completed and files have been transferred to the local machine

Application mpiBLAST

- mpiBLAST - a tool used for sequence analysis and interpretation in genomic sequencing
- Assumptions:
  - mpiBLAST installed on resource
  - Data stored on resource
  - Database pre-distributed

Future work

- Create and store user and application profiles in database for easy retrieval
- Use this knowledge to help in estimating time-to-completion as well as to adjust fuzzy logic parameters
- Provide set of API to enable extension
- Incorporate Job Description Submission Language (JSDL) standard once available

Summary

- Emerging grid technologies are promising, but usability may present a problem when transitioning to the Grid
- Provide user friendly access
- Use given application performance for load prediction and resource selection

Questions?

Contact me at: afgane@uab.edu

Slides available at: www.cis.uab.edu/afgane