βtest- GridCom-TC v1.0

Grid Computing Training Courseware

Grid Computing Training Courseware *v*-1.0

Designed for Testing, Benchmarking & Performance Activities

Document Title	**Grid Computing Training Courseware (βtest-GridCom-TC-*v*1.0)**
Users	Betatesting Group Members of NPSF, C-DAC,Pune
Source	Parallel Computing /Grid Computing conducted by Betatesting group, C-DAC, Pune & Author's Experience on Processing and Grid Computing Projects
Date:	August 01, 2006
No. of Pages	41
Author	VCV.Rao
Contact	betatest@cdac.in

β-Testing Group, C-DAC

Betatesting Group, National PARAM Supercomputing Facility,
Centre for Development of Advanced Computing (C-DAC)

Non-Disclosure

The information in this proposal shall not be disclosed outside the intended party and its relevant stakeholders and shall not be duplicated, used, or disclosed in whole or in part for any purpose other than to evaluate the proposal, provided that if a contract is awarded to C-DAC as a result of or in connection with the submission of this proposal, the client shall have the right to duplicate, use, or disclose the information to the extent provided by the contract. This restriction does not limit the right of the client to use information contained in the proposal if it is obtained from another source without restriction.

Contents:

1. Objective…… 1
2. An overview of βtest-GridCom-TC -v1.0 Courseware…… 2
3. Time Duration……..…… 6
4. Description of various Modules. ……..……..…… 7
5. References……..... 25

1. Objective

The aim is to design Grid Computing courseware and we focus on many of the issues that an architect or developer or tester needs to be aware of when designing a grid-enabled application as well as performance of a grid-enabled application.

The training courseware objective is to build expertise for execution of *Test plan and Benchmarks for grid infrastructure* with focus on the open source Globus Toolkit 2.x/4.x. The foundation is focused on quickly adapt to developments in enabling applications to grid environment.

The courseware is useful for beginners, middle level and advanced level grid users. The course has been classified into two-tier level, focusing on various aspects of Grid infrastructure and each tier module gives an overview of topics, which benefit the on-going project activities.

The complete courseware forms a single concentrated course on grid computing, which is a continuously evolving resource at C-DAC. The courseware can be easily tailored to the developer, testing group, user community to extract performance of large scale applications and gives a strong foundation on programming models for Benchmarking computing systems of grid infrastructure.

The philosophy is to introduce new functionality and concepts to solve a design, implementation or analysis of enabling applications for Grid Computing with Globus in this courseware. Various parts of the courseware and details of modules are summarized below.

To understand about the sorts of problems, beginners, developers encounter, when they begin thinking in write grid programming programs using different programming paradigms, the course contents may help and it is designed for short, and long-term duration of time schedule.

Suggested below are one ***short-term*** and ***long-term*** courses or preliminary/exhaustive training programe of grid programmers. Some modules may focus on theory and laboratory session as per requirements of project activities. However, some groups in C-DAC may require detailed contents of modules, pertaining to the on-going research projects.

For specific project work, it is necessary for the course participant to refer advanced books on Grid Computing or visit important web sites.

2. Various part of βtest-GridCom-TC v1.0 Courseware

The βtest-GridCom-TC *v*1.0 courseware can be grouped into *twelve* parts and each part has sufficient number of modules. Each module contents can be covered in classroom lectures and the Hands-on Session can be done on grid laboratory. A brief summary of various parts and flow of several modules of courseware is discussed below.

Part I - Introduction to Grid Computing: Reasons for Computational Grids, Some definitions, Grid Applications Classes An overview of Using Grids – Grid Developers, Tool Developers, Grid Programming Environment, Application Developers, End Users, System Administrators, Major Grid Projects

Part II - Application Requirements & Characteristics: Category of applications - First-in-time, or an on-demand computing, Data-Intensive Computing, Collaborative Computing, High Throughput Computing, Distributed Supercomputing, A Grid Systems Taxonomy - Design Objectives of Applications - (Computational Grid, Data Grid, Service Grid); Classification of Grid Applications – Class-I(Loosely Coupled), Class-II (pipelined), Class-III (Tightly Synchronized), Class-IV (Widely Distributed); Performance Metrics; Reproducibility of results, Measurement of Turnaround time and Throughput, Quantification of Grid middleware Overheads, The ratio of communication and comp., Use of numerical libraries on computing systems of different sites with architectural diversity, Algorithms to handle latency and bandwidth requirements of application

Part III-Globus 2.4 Services: An Overview of Globus Toolkit- 2.4; Resource Management, Layered Grid Architecture- Protocols and Services and API; The Hour Glass Model of Globus Architecture, Configure Globus Toolkit; An overview of GRAM, Installation and Set-up of Globus 2.4, Data Management GRIDFTP, Monitoring and Discovery (MDS), Grid Security Principles (GSI)

Part IV-Globus 4.0 Services: GT4 Architecture Overview, Layered Grid Architecture- Protocols and Services and API Installation and set-up of Globus 4.0, GT4- Distributed systems and Web Services, OGSI Implementation, Open grid Service Architecture (OGSA), GT4 Service Oriented Applications and Infrastructures, GT4 Web Services Implementation, GT4 Web Service Specifications, GT4 Service Oriented Architecture; Grid Security Principles (GSI)

Part V-Application Architecture Considerations: The Characteristics of grid application, Understand Globus 2.4 Components- GSI, GRAM, MDS, GridFTP, GSI, GASS, Grid Programming language considerations; Characteristics of application flow and job flow; Job Dependencies on System environment; Job topology, Job Criteria (Standard application, Parallel Applications); Passing of data input/output; Qualification scheme for grid applications, Knock-out criteria for gird applications

Part VI-Grid Data Management Considerations: Grid Programming Model – Shared data and shared nothing, Data Characteristics, Data Management techniques and solutions -Shared file system, Replication, Caching effects, Data Access Control Systems, Database solutions for grids, Distributed Data approaches, Replication, Global Parallel File System (GPFS) Data Management techniques

Part VII-Grid Programming Environment -I: The Programming Problem; Compilers languages and Libraries; Grid Global Compilation Systems, Grid Programming Model – Shared data and shared nothing, Object Oriented-based applications; Examples of Grid Programming Model, Grid Web Services and Peer to Peer (P2P) Grid Concepts, scripting languages (Javascripts, Perl, Python, Tcl/Tk, Unix Shells)

Part VIII- Grid Programming Environment -II: Middleware & Tools: Network Enabled servers, Frameworks and Component based technology, Grid enabled RPC Systems, Middleware Systems – Legion, and Object Oriented Technology, XML technology, Services & Protocols (Web Services/XML), Scripting language, Problem Solving Environments, Portals, Application Specific tools

Part IX- Enabling Applications for Grid Computing: Write C/C++, Perl COG programs using Globus APIs *globusjobrun* command and associated APIs, and Resource Specification Language (RSL) scripts for submitting jobs, Write JAVA CoG programs using Globus APIs *globusjobrun* commands and associated APIs, Example programs using MPICH-G2 & C/C++ language; programs on Perl CoG, Python CoG using Globus APIs; Distributed Applications using Common Component Architecture technologies (XML, Web-services) and Grid Services

Part X- Grid Integration Test Scripts: Automatic Test scripts for Grid with Globus Toolkit – Interoperability Test Suites – RSL-Hello World, Stage, RSL-Shell, BatchJob Submitt, BatchJob Query, GASS, GridFTP, GSIssh, GSIscp tests; Grid MDS Tests – GRIS, Job managers, Support of *globusrun* job commands

Part XI- Low Level Grid Benchmarks (Grid Probes): Estimate Basic authentication for all grid nodes, query MDS, Ping, Ping-Pong, Sleep, Data Transfer, Circular, and Gather Probes, Compute and Communication Benchmarks

Part XII- Application Grid Benchmarks: NAS Grid Benchmarks; GridBench (CrossGRID) - Micro, Macro and Application Benchmarks and Performance Evaluation on Grid (Resources, Site Configuration), Benchmark Metrics, and Grid Bench Definition Language.

Most importantly, the courseware has sufficient number of programming assignments, which should play a central role to make strong foundations on grid programming in order to solve particular applications. Several modules defined in the courseware can be grouped together as per requirements of members who opt for short, and long term time duration.

Figure 1 illustrates the various parts of the courseware and Figure 2 illustrates the various modules of each part and its relations with other modules. The solid arrow from Module *A* to Module *B* indicates Module *B* depends heavily upon material presented in Module *A*.

For long-term courseware schedule, Module *A* and Module *B* have mutual relations, while performance of application programs with programming environment are considered on target architecture.

All course contents given in each part is covered in numerical order, you will satisfy all requirements for Testing, Benchmarks execution, and enabling applications for grid computing environment with Globus. However, you would like your members to start programming in *C* or *C++*, or *Perl* or *Phyton*, or *Java* or with Globus as quickly as possible, you may wish to skip some modules in Part IX or cover only two or three modules of the courseware. Definitely there is weak dependency across several modules of different parts and judicious choice should be taken up, which merely depends upon the duration of courseware i.e. Short, or Long term.

If you wish to focus on expertise of Globus computing infrastructure, you must work on Part-I, III, IV and you may skip Part IX.

If you would like to start by having your members programming examples with focus on Benchmarks and performance, you can jump to Part IX, XI, XII) after covering important modules in remaining parts.

To get exposure to new functionality ‘just in time’ and enabling applications to Grid Computing with Globus ToolKit, one can jump to various modules in Part I, III, IV, VII, VIII & IX with strong foundation on Part III & IX. A pre-requisite for this is to go through all modules as defined in short term course.

Most importantly, if you wish to get expertise on Testing of Grid Infrastructure with Globus Toolkit, you can work on Part XI immediately after completion of necessary modules in other parts. Special emphasis on solving complex real-life applications on Grid infrastructure and enabling applications for Grid Computing with Globus Toolkit is discussed, with focus on application requirements & Characteristics as described in Part-II & Part V. The topics in the modules of Part-II may assist the grid programmer to identify application requirements on grid infrastructure.

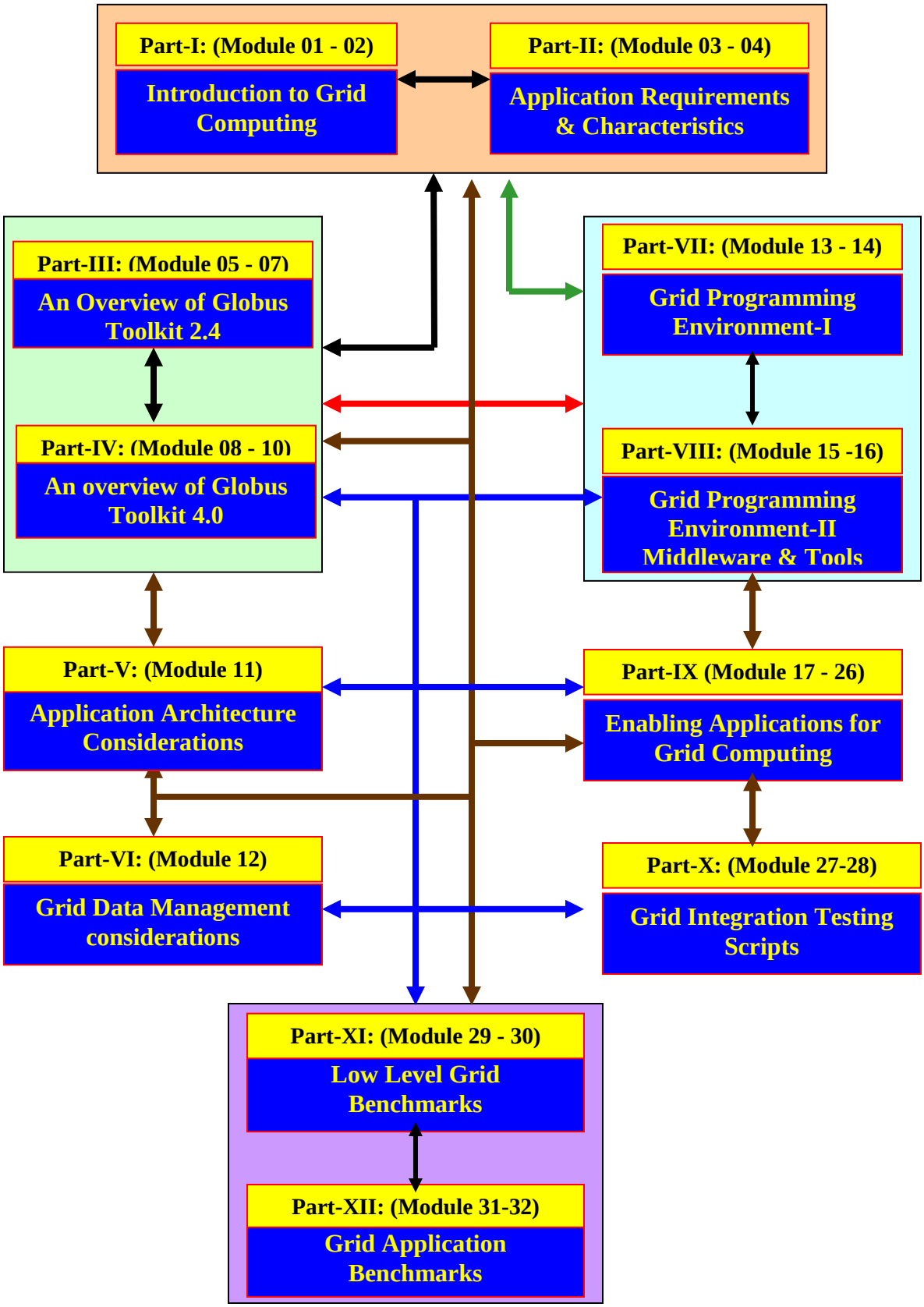

Fig 1. Flow representation of Grid Computing courseware Parts/Modules

3. Time Duration

Suggested below is time duration for courseware program and weightage may vary as per the requirements of project. The Hands-on on some modules is compulsory and it can be conducted on grid laboratory.

- Tier - 1: Short term course and the time duration is 60 Calendar days
- Tier - 2: Long term course and the time duration is 90 Calendar days.

The **short-term** course is focused on identifying suitable modules from Part-I to Part-X and quickly learns to write grid programs with focus on different programming languages using Globus Toolkit. The programming samples developed in this courseware provide the basic techniques required to get you started in the application development for grid environments. The applicability of grid environment to the particular application is important.

The **long-term** course provides complete one semester course in grid programming, focusing on turnkey projects. The topics such as grid infrastructure considerations, application specific considerations, and data management considerations, choice of appropriate grid programming environment and tools are considered. Enabling applications for grid computing with Globus Toolkit require in depth study of these topics and this can be done in an incremental fashion.

4. Brief Summary of various parts of Grid Computing courseware

Details of various parts of courseware and an overview of module topics are given below.

Part-I: An Introduction to Grid Computing

Module Names : Module 1(GridIntro-01); Module 2 (GridIntro-02)
Project Type : Short /Long term
Time Duration : 7 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand basic concepts of Grid Computing; Grid Infrastructure considerations, Grid Programming considerations Examples of real-life applications; Deliver lectures, Participate in Group discussions

Modules 1- 2 **Calendar Days: 7**

Introduction to Grid Computing, An Overview of Grid Computing, Types of Grids (Computational, Data, Services), Reasons for Computational Grids, Some definitions, Grid Applications Classes; An overview of Using Grids – Grid Developers, Tool Developers, Grid Programming Environment, Application Developers, End Users, System Administrators, An Overview of Globus Project, An Overview of Globus Components (Portals; Security, Broker, Scheduler, Data management, Job and resource management, Job flow in grid environment); Globus toolkit 4.0 and the Open grid Services Architecture (OGSA), The Open grid Service Interface (OGSI), An overview of Grid Infrastructure components, Major Grid Projects

Part-II: Grid Application Requirements & Characteristics

Module Names : Module 3(GridAppln-01); Module 4 (GridAppln-02)
Project Type : Short /Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand requirements of Grid Applications, Study different algorithms for real-life applications; Understand Programming paradigms of Distributed Computing systems, Deliver lectures, Participate in group discussion

Module 3 - 4 **Calendar Days: 14**

Category of applications - First-in-time, or an on-demand computing, Data-Intensive Computing, Collaborative Computing, High Throughput Computing, Distributed Supercomputing, *A Grid Systems Taxonomy* - Design Objectives of Applications - (Computational Grid, Data Grid, Service Grid); *Classification of Grid Applications* – Class-I (Loosely Coupled), Class-II (pipelined), Class-III (Tightly Synchronized), Class-IV (Widely Distributed); *Performance Metrics*, Reproducibility of results, *Global Compilation system for Applications*, *Performance of Applications* (Measurement of Turnaround time and Throughput, Quantification of Grid middleware Overheads, The ratio of communication and computation), Use of numerical libraries on computing systems with architectural diversity, Algorithms to handle latency and bandwidth requirements; Data Intensive application Characteristics and Replica Management of Data, Application flow, Programming language considerations, system environment

Part-III: Globus Toolkit Version 2.x

Module Names : Module 5(Glbous2-1); Module 6 (Globus2-2); Module 7 (Glbous2-3)
Project Type : Short /Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Globus 2.4; Installation & Setup of Globus2.4 on Heterogeneous Systems; Understand Globus 2.4 Components- GSI, GRAM, MDS, GridFTP, GSI, GASS, Deliver lectures, Deliver lectures on Globus Group discussion

Module 5 - 7 **Calendar Days 14**

An Overview of Globus Toolkit- 2.4; Resource Management, Layered Grid Architecture- Protocols and Services and API; The Hour Glass Model of Globus2.4 Architecture, Configure, installation, & Set-up Globus 2.4 Toolkit; Grid Security (GSI) – Authentication, Authorization, Grid secure communication, Other grid communication, Application enablement considerations – Security, An overview of Globus Resource Allocation manager (GRAM) – the *globusrun* command and associated APIs, Resource Specification Language (RSL), The gatekeeper daemon, The job Manager, Dynamically-Updated Request Online Coallocator (DUROC), Information Services -Monitoring and Discovery Service (MDS), the Data Management Services - GridFTP, Global Access to Secondary Storage (GASS), data transfer utilities, Scheduler, Broker, Inter-process communications (IPC), Grid Portal and Integration of application with grid-portal, Performance, Reliability, topology Considerations, mixed platform grid environments.

Part-IV: Globus Toolkit Version 4.0

Module Names : Module 8(Glbous4-1); Module 9(Globus4-2); Module 10(Glbous4-3)
Project Type : Short/Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Globus 4.0; Installation & Setup of Globus 4.0 on Heterogeneous Systems; Understand Globus 4.0 Components- GSI, GRAM, MDS, GridFTP, GSI, GASS, Deliver lectures on Globus ToolKit; Understand WSRF – Web Services; Group discussion

Module 8 - 10 **Calendar Days 14**

GT4 Architecture Overview, Layered Grid Architecture- Protocols and Services and API Installation and set-up of Globus 4.0, GT4 Service Oriented Architecture, GT4-Distributed systems and Web Services, Overview of Changes from GT2 to GT4; GT4 Service Oriented Applications and Infrastructures, OGSI implementation, Open Grid Service Architecture (OGSA), GT4 Web Services Implementation, GT4 Web Service Specifications (XML, SOAP, WSDL, WSRF), GRAM Overview – Use of WS-Resource Mechanisms, Data Management – GridFTP, Reliable file Transfer Service (RFT), Replica Location Service (RLS), Monitoring and Discovery – MDS Index services, GT 4.0 Security Principles, Web Services Authentication and Authorization, GSI OpenSSH, User Interfaces and tools, Inter-process communications (IPC), Grid Portal and Integration of application with grid-portal, Performance, Reliability, Topology Considerations, Mixed platform grid environments.

Part-V: Grid Application Characteristics Considerations

Module Names : Module 11(GridApplnChar-1)
Project Type : Short/Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : The Characteristics of grid application, Understand Globus 2.4 Components- GSI, GRAM, MDS, GridFTP, GSI, GASS, Grid Programming language considerations; Characteristics of application flow and job flow; Job Dependencies on System environment; Job topology, Job Criteria (Standard application, Parallel Applications); Passing of data input/output

Module 11 **Calendar Days 14**

The Characteristics of grid application and job, Application flow in a grid, Characteristics of application flow and job flow; Parallel flow (Several jobs in parallel), Input/output Characteristics – Data Producer and Consumer, Serial application flow (sequential job on one processor of site), Pipelined flow job characteristics, Network flow for different jobs, Jobs and sub-jobs for given application, Job Criteria – Number of jobs in parallel (Batch job, Standard Application, Parallel Application, Interactive job), Programming language considerations, Job Dependency on System environment, Check-pointing (launching of same jobs at different points of time), Job Topology – architectural characteristics of job, intra-gird, inter-grid); Passing data input/Output/Stage back, Usability requirements for grid solution, Reliability of the grid application, Qualification scheme for grid applications

Part-VI: Grid Data Management Considerations

Module Names : Module 12(GridDataMange-1)
Project Type : Short/Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Globus 2.4 Components; Grid Programming language considerations; Data Characteristics, Data Management techniques and solutions -Shared file system, Replication, Caching effects, Data Access Control Systems, Database solutions for grids, Distributed Data approaches, Global Parallel File System (GPFS)

Module: 12 **Calendar Days 14**

An overview of Data used (distribution and location) in the Application, Individual and separated data per job features, Shared data access, Read, Write Databases and Locking databases, Size of Data, Network Bandwidth, Time sensitive of Data, Data topology per job, Data Topology Graph, Data Types, Data Volume and grid scalability, Data Distribution, Encrypted Data, Data Management techniques, Shared file system, Databases, Replication (Distribution of files across a set of nodes), Cache Mechanism, Role of Transfer agents in Data Transfer, Peer-to-Peer data Transfer, Global file system approach – Global parallel File System (GPFS), Network File System (NFS), The Global File System (GFS), Replica Catalog, Replica Location Service (RLS), Database solutions for grids, Federated databases, OGSA database Access, Data Brokering, Data grid projects

Part-VII: Grid Programming Environment-I (Prog. Languages)

Module Names : Module 13(GridProgEnv-1); Module 14 (GridProgEnv-2);
Project Type : Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Globus 2.4; Installation & Setup of Globus2.4 on Heterogeneous Systems; Understand Globus 2.4 Components- GSI, GRAM, MDS, GridFTP, GSI, GASS, Grid Programming language considerations; Job Dependencies on System environment; Job topology, Job Criteria (Standard application, Parallel Applications); Passing of data input/output

Module 13 - 14 **Calendar Days 14**

The Programming Problem – Compilers, languages and Libraries; Grid Global Compilation Systems, Grid-enabled implementation of MPI *v*1.1standarad using Globus Toolkit, Distributed Virtual shared memory model (Grid targeted DVSM), Message Passing using RPC mode, scripting languages (Javascript, Perl, Python, Tcl/Tk, Unix Shells), Object Oriented-Based applications; Programming and building C/C++ applications, JAVA CoG Kit using Globus Toolkit, Common Component Architecture and Component based technology (CBT), Services, Web Services Description Language (WSDL), XML technology, Examples of Grid Programming Model, Grid enabled programming tools (JAVA CoG Kit, Python, EJB, Perl, and CORBA) using Globus Toolkit, Peer to Peer (P2P) Grid Concepts

Part-VIII:Grid Programming Environment-II(Middleware & Tools)

Module Names : Module 15(GridProgEnv-4), Module 16 (GridProgEnv-5),
Project Type : Short/Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Installation & Setup of Globus2.4 on Heterogeneous Systems; Understand Globus 2.4 Components; Grid Programming language considerations; Install and Demonstrate selected third party Tools, Deliver lectures, Develop example programs using tools

Module: 15 **Calendar Days 14**

The Grid Middleware Tools: Client-server systems (Network-enabled servers *Ex: NetSolve*, *Ninf*), GridRPC based systems; Frameworks and Problem Solving Environments (Ex: Cactus –) Parallel file I/O operations, PETSc Scientific library, adaptive mesh refinement, web interfaces and advanced visualization tools) Metachaos (interoperability between various data parallel runtime libraries); framework for Distributed Computing Applications (Ex: ASSIST), Component based technology (CBT) - a higher-level Grid Programming, CORBA, DCOM-Microsoft, Enterprise Java Beans (EJB Sun), .NET (Microsoft), Component based technology (CBT), Web Services (XML, UDDI, SOAP, WSDL) and Grid Services, Open Grid Services Architecture (OGSA), XML technology, Grid enabled programming tools (JAVA CoG Kit, Python, EJB, Perl, and CORBA) using Globus Toolkit, Scripting languages

Module: 16	Calendar Days 4

Portals and Problem Solving Environment: Building a simple web portal, programming Language used in portal code (Ex. Java) Integrating web portal function with a grid application, Add methods to execute the Globus commands (Running commands Using Java Code), Security issues, Submit grid application for execution (Get Application Status,), Using Java CoG kit and utilize basic Grid services, Use Java Client Kit which provides client side access to many grid services, Integration of more advanced Grid Services into the CoG Kits. An overview of Python CoG Kit, Perl CoG Kit, CORBA; An overview of WebFlow Project – Java-based three tier architecture uses Globus toolkit services; The Grid Portal Development Kit leverages off existing Globus/Grid middleware infrastructure as well as commodity web technology including Java Server Pages and servlet; An overview of Legion Portal (Perl, PHP, MySQL and the Common Gateway Interface (CGI) mechanism for involving Legion commands); NPACI Grid Computing portal; An overview of HotPage – NPACI User Portal

Part-IX: Enabling Applications for Grid Computing using *Globus job submission* commands

Module Names : Module 17(GridProgApp-1)
Project Type : Short term
Time Duration : 7 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Globus 2.4 Environment and Components; Grid Prog. Language considerations; Develop and Demonstrate Example programs using Globus APIs; Deliver Lectures

Module: 17	Calendar Days 7

Write simple example programs using components of Globus APIs. The examples are based on the GRAM for submitting jobs, and the MDS for finding resources.

1. Generate Globus Makefile
2. Write Program to write the globus_module API functions to activate and de-activate the modules
3. Write simple programs using *Globus Toolkit* Shell Commands that can be evoked by an application
   – globus-job-run (launch a simple job)
   – globus-job-run (submit a multi-request-query)
   – *globus-job-run* (example on globus-job-run-example; example on multi-request-query)
   – *globus-job-submitt* **(**submits job in the background)
   – Retrieve information about a job; cancel the job, to clear the jobs produced by a job
   – globus-job-run example using RSL Script
   – globus-job-submitt example using RSL Script
   – Generate RSL using globus-job-submitt-dumprsl
4. Write simple programs using *Globus Toolkit* Shell Commands and GSIssh that can be evoked by an application
   – gssissh example (check for grid-proxy-init)
   – Running a graphical application through 'gsissh' - User to remotely run a application that will be displayed on local server.
   – Gsissh example proxy delegation support (A user connects to site'A' and from there can submit a job without the need to regenerate a new globus proxy).

Part-IX: Enabling Applications for Grid Computing using Globus job submission commands & RSL

Module Names : Module 18(GridProgApp-2)
Project Type : Short term
Time Duration : 7 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Globus 2.4 Components- GRAM, GridFTP, GSI,

Module: 18 **Calendar Days 7**

Write programs using Globus ToolKit APIs and Resource Specification Language (RSL), which are used by clients to specify the requirements of the job by user.

1. RSL script – executes the `/bin/ps -ef` command and stores the standard output of the script in `/tmp/temp2`
2. RSL script to Execute 5 instances of *p1* on a machine with at least 64MB of memory and execute *p2* on a machine with an ATM connection
3. RSL script to create 5 instances of *myprog*, each on a machine with at least 64 MB memory that is available for 4 hours
4. RSL script: Create a proxy
5. RSL script – choosing the appropriate resources; Multiple sub-jobs; Accessing job results; Job Management (Canceling the job)
6. RSL script: Retrieving credentials from an existing proxy.
7. RSL script: using MaxMemory; MinMemory; maxtime, maxwallTime, maxcpuTime
8. RSL script to execute the resource manager name using GRAM
9. RSL script to get information about grid resources accessing MDS
10. RSL script to display the currently executing processes using the `ps` command
11. RSL script: Use *rsl_substitution* to create variables within the RSL String
12. RSL script to a simple job to a known resource manager
13. RSL script to run the executable at remote site and stage back the output file to client;
14. RSL script to transfer the file from client site to remote site;
15. RSL script to transfer the file from remote *site A* to remote *site B.*
16. RSL script to execute a job at remote *site A* with executable file at *site B* and stage back the output file to client;
17. RSL script to execute job at remote *site A* with executable file at *site B,* Input file from site C and stage back the output to the client
18. RSL script to launch a job at Site A and transfer file from Site A to Site B, and Site B to Site C and so on ``… in a ring fashion and stage back the output to the client.
19. RSL script to launch a job at Site A and transfer two files from Site A to Site B, and Site B executes matrix into matrix multiplication algorithm with input as received data files from Site A and stage back the output to the client.
20. RSL script to launch a job at Site A and transfer data file to '*p*' Sites. Each site performs computation and the output is send back to to *Site A. Site A* gathers the output from all sites and transfers the new data file to 'p' sites.. The process continues for given iterations and the *Site A* terminates the job and sends the output information (CPU time, memory used, Compiler, OS information used by each Site) to the Client.

Part-IX: Enabling Applications for Grid Computing using Globus job submission commands & C/C++ languages

Module Names : Module 19 (GridProgApp-3)
Project Type : Short /Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Globus 2.4 Components- GRAM, GridFTP, GSI, GASS, Write programs to submit a job in a C or C++ program and RSL string describing the job

Module: 19 Calendar Days 14

Write programs to submit a job in a C or C++ program and RSL string describing the job must be provided using components of Globus APIs. Globus_gram_client API provides an easy API for job submission. The Blocking and Non-blocking calls can be used.

1. Write a globus-based "*hello, world*" C -program using Globus APIs
2. Write a globus-based "*hello, world*" C++ -program using Globus APIs
3. To test fully qualified domain names (FQDN) using *globus_libc_gethostname library call*
4. C or C++ *program for Job submission* using ITSO_GRAM_JOB class and example of a callback to the globus_gram_client_callback_allow() function; (globus_gram_client_callback_allow() callback function; globus_gram_client_register_job_request_call; GRAM job submission via an ITSO_GRAM_JOB_object;
5. Using ITSO_GRAM_JOB, submit a job, check if it has failed, and ,if so submitt again to another node. (Use the function globus_gram_client_ping( ))
6. Broker Examples: Use the LDAP API provided by the Globus tool-kit to send the request to the main GIIS server location.
7. Broker Examples: Use the LDAP API provided by the Globus tool-kit and submitting the LDAP query
8. Broker Examples: Use the LDAP API provided by the Globus tool-kit and retrieving results of the submitted query from Globus MDS
9. Broker algorithm implementation
10. Application that takes the first argument as the number of required nodes running the Linux operating system (Use GetLinuxNodes() to get *n* nodes)

Part-IX: Enabling Applications for Grid Computing using Globus job submission commands, RSL & Perl CoG

Module Names : Module 20 (GridProgApp-4)
Project Type : Short/Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Globus 2.4 Components- GRAM, GridFTP, GSI, GASS, Write programming examples for Globus using Perl Lang.

Module: 20 Calendar Days 14

Write programs to submit a job in a Perl CoG program and RSL string describing the job must be provided using components of Globus APIs. Globus_gram_client API provides an easy API for job submission.

1. Write *Perl* script to test TCP connection establishment in a typical grid environment.
2. Write a Perl program for *proxy_check* using Globus APIs
3. Write a Perl program for *GRAM authenticaiton* using Globus APIs
4. Write a Perl program to calculate the bandwidth to transfer 1 MB to 100 MB data from *host site* to *remote site* using Globus APIs
5. Write a Perl program to calculate the *latency* time for data transfer from *host site* to *remote site* using Globus APIs
6. Write a *Perl* program to *print summary* of the output generated in problem (3) & (4).
7. Write a *Perl* program to read the input data file from *host site* and *transfer* the data to *remote* site.
8. Write a *Perl* program to verify *source file existence* and *remote directory space* as per requirements of example when the data is transferred from *host* to *remote* site.
9. Write a *Perl* program to handle the *STDOUT* and *STDERR* messages on *host* site or *remote directory space* as per requirements of application when the data is transferred from *host* to *remote* site.
10. Write a *Perl* program to display the results of data transfer from *host* site to *remote* site in which 10 MB of data is transferred in first iteration and an increment of 20 MB for subsequent iterations for 100 iterations, using the public domain software GNU plot.
11. Write a *Perl* program in which the data is *transferred* from Site A to Site B, and site B to Site C and so on in a *circular* fashion.
12. Write a simple *Perl* program pseudocode example of using the Perl CoG Kit's Globus::GlobusJob and Globus::RSL modules which shows assembly of an RSL string describing a job, submission of the job, job status discovery, and collection the *stderr* and *stdout* from the job.
13. Write a simple *Perl* example program of using the *Perl CoG Kit's Info::Search* module to perform an MDS query, then print out the key/value data retrieved from the query.
14. Write an Perl example program of using the GSI::Proxy module to create a proxy certificate from an X.509 certificate and key.
15. Write a *Perl* example program of using the GSI::Myproxy module to store a proxy in a Myproxy server, then retrieving a delegated proxy from that Myproxy server.

Part-IX: Enabling Applications for Grid Computing using Globus job submission commands, RSL & Python CoG

Module Names : Module 21 (GridProgApp-5)
Project Type : Short/Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Globus 2.4 Components- GRAM, GridFTP, GSI, GASS. Write programming examples for Globus using Python Lang.

Module: 21 **Calendar Days 14**

Write programs to submit a job in a Python (Python COG) program and RSL string describing the job must be provided using components of Globus APIs.

16. Write Python script to test TCP connection establishment in a typical grid environment.
17. Write a Python program for *proxy_check* using Globus APIs
18. Write a Python program for *GRAM authenticaiton* using Globus APIs
19. Write a Python program to calculate the bandwidth to transfer 1 MB to 100 MB data from *host site* to *remote site* using Globus APIs
20. Write a Python program to calculate the *latency* time for data transfer from *host site* to *remote site* using Globus APIs
21. Write a Python program to *print summary* of the output generated in problem (3) & (4).
22. Write a Python program to read the input data file from *host site* and *transfer* the data to *remote* site.
23. Write a Python program to verify *source file existence* and *remote directory space* as per requirements of example when the data is transferred from *host* to *remote* site.
24. Write a Python program to handle the *STDOUT* and *STDERR* messages on *host* site or *remote directory space* as per requirements of applcation when the data is transferred from *host* to *remote* site.
25. Write a Python program to display the results of data transfer from *host* site to *remote* site in which 10 MB of data is transferred in first iteration and an increment of 20 MB for subsequent iterations for 100 iterations, using the public domain software GNU plot.
26. Write a Python program in which the data is *transferred* from Site A to Site B, and site B to Site C and so on in a *circular* fashion.
27. Write a simple Python pseudocode example of using the Python CoG Kit's Globus::GlobusJob and Globus::RSL modules which shows assembly of an RSL string describing a job, submission of the job, job status discovery, and collection the stderr and stdout from the job.
28. Write a simple Python example program of using the Python CoG Kit's Info::Search module to perform an MDS query, then print out the key/value data retrieved from the query.
29. Write an Python example program of using the GSI::*Proxy* module to create a proxy certificate from an X.509 certificate and key.
30. Write a Python example program of using the GSI::*Myproxy* module to store a proxy in a *Myproxy* server, then retrieving a delegated proxy from that *Myproxy* server.

Part-IX: Enabling Applications for Grid Computing using Globus job submission commands, GridFTP & JAVA CoG

Module Names : Module 22 (GridProgApp-6)
Project Type : Short /Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Globus 2.4 Components- GRAM, GridFTP, GSI, GASS. Write programming examples for Globus using Java

Module: 22 Calendar Days 14

Write programming examples for Globus using Java CoG (The examples in Java for interfacing with Globus components and functions).

1. GSI/Proxy: Create a proxy compatible with Globus Toolkit 2.4 (grid-proxy-init)
2. GSI/Proxy: Create a proxy compatible with Globus Toolkit 2.4 (The CA public key can be loaded using the using the CertUtilis class,Create a new certificate using your private key)
3. GSI/Proxy: Retrieving credentials from an existing proxy
4. GSI/Proxy: Destroying the proxy
5. GRAM: Submit a simple job to a known resource manager (use RSL String)
6. GRAM: Check, if you are allowed to submit a job to specific resource manager
7. Accessing MDS: Example for import statements and variable declarations for the MyGridInfoSearch Class (GridinfoSearch Class provided in the org.globus.tools.package for the JavaCogKit);
8. RSL example (parse and display the RSL String; Store the RSL string as type RslAttributes; The RslAttribute class allows parsing, modifying, and deleting value in the string;)
9. RSL script: Find the value of a specified attribute using The *getSingle( )* method
10. RSL script: Remove given attribute from the RSL string and the remaining string is printed to the screen (Use *remove( )* method)
11. RSL script: Adds a value "GARUDA" to the attribute argument. (Use *add( )* method)
12. RSL script: Get the value for the specified variable (Use *get( )* method)
13. RSL script: Get all the variables declared within rsl_substitution(Use
14. *getVariables( )* method)
15. RSL script: Create attributes using ListRslNode class and use *add( )* method to create a new RSL string

Part-IX: Enabling Applications for Grid Computing using Globus job submission commands, GridFTP, GASS,RSL & JAVA CoG

Module Names : Module 22(GridProgApp-7)
Project Type : Short /Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Globus 2.4 Components- GRAM, GridFTP, GSI, GASS, Write programming examples for Globus using Java CoG

Module: 22 **Calendar Days 14**

Write programming examples for Globus using Java (The examples in Java for interfacing with Globus components and functions).

1. GridFTP: Perform a third party file transfer (Using extended block mode and GSI security using the GridFTP protocol – Understand functionality of Client /Server FTP file transfer & create client on each server. Read credentials from the proxy file and for authentication GSSCredential object .Use the GridFTP port, authenticate to GridFTPClient first, before setting or changing any other properties like transfer-type or mode; Define the source and target files)
2. GridFTP Client-Server: GridFTP Client-server example (Local interface to the storage must be supplied to Server /Client; Toolkit provides two interfaces: DataSink for receiving a file, and dataSource for sending a file; Setting up the client mansually to active or passive is possible, but not required for third-party transfers)
3. URL: The URL Copy Example (The URL Copy class provides a very easy way of transferring files. It understands the GSIFTP, GASS, FTP, and FILE protocol .URL object properties like DestinationUrl and SourceAuthorization, For third party transfer, use ucopy.setUseThirdPartyCopy (true))
4. GASS API: Send data from Site 1 to Site 2 and stage back to Client; Send application output on Site 1 to Site 2 and final result to Client
5. GASS API: Batch GASS: Submit a job and retrieve the results
6. GASS API: Interactive GASS Example: Submit a job and retrieve the results

Part-IX: Enabling Applications for Grid Computing using Globus and MPICH-G2

Module Names : Module 23(GridProgApp-8)
Project Type : Short /Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Installation & Setup of Globus4.0; Globus4.0 Components- GRAM, GridFTP, GSI, GASS, globus_io, Write RSL files for MPI jobs and execute MPI jobs

Module: 23 Calendar Days 14

Write MPICH-G2 (MPI based C or C++ programming examples) for Globus using RSL and other associated Globus APIs

1. Simple program for testing vendor-supplied *MPI mpirun*'s ability to export environment of MPICH-G2
2. Use *mpirun* by supplying your own RSL script
3. How to write your own RSL script by using mpirun (Specifying `-dumprsl` on the *mpirun* command prints the generated RSL)
4. Example programme *hello.c* & use *hello.rsl* and *globusrun* for running "hello world" under *Globus 2.4*
5. Write simple example "Hello_world" program using *MPI_Attr_put* library call and *Globus_Grid_FTP* library.
6. Example program for testing vendor-supplied MPI mpirun environment and "hello, World" using RSL.
7. MPI program to send large data of size more than 500 MB from one process to another process using Parallel TCP Streams over point-to-point TCP channels between two processes. (*MPICH-G2* – threaded flavours of *Globus* is must)
8. Write MPI Client/Server application using MPI_Open_Port, MPI_Cloise_port, MPI_Comm_accept and MPI_Comm_connect
9. Write Perl script to test TCP connection establishment and communication in client and server environment
10. Write RSL script to execute an application on *three* sties in which *two* sites are equipped with vendor supplied MPI and the *third* site is not. (Assume that fictitious binary-compatible machines have access to the same file system). The application uses *MPI_Alltoall* library call & input data of 500 MB single precision real values are randomly generated on each process and the program is executed on *three* sites with total of *five* processes. (MPICH-G2 selects five processes)
11. Write the RSL script to execute *MPI_Alltoall* library call application on five sites using one processor at each site. You have to select the order of sites as per communication performance of Grid Architecture. Use *mpirun* by supplying your own RSL script and assume the input data as explained in problem 8
12. Write the RSL script to execute *MPI_Alltoall* library call and measure the communication time with input message size on every process, starting with 10 MB to 200 MB with an increment of 10 MB on 5 processes.

Part-IX: Enabling Applications for Grid Computing using Globus and MPICH-G2

Module Names : Module 23(GridProgApp-8)
Project Type : Short /Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Installation & Setup of Globus4.0; Globus4.0 Components- GRAM, GridFTP, GSI, GASS, globus_io, Write RSL files for MPI jobs and execute MPI jobs

Module: 23 **Calendar Days 14**

13. Write RSL file and execute " MPI gather" program as a single job across three sites (each processes at each site). Each process generates its own data of size 100 MB and process with rank 0 accumulates the data and stage back to the client.
14. Write RSL file to run "broadcast" MPI program as a single job across six sites (two process at each site) and "tell" MPICH-G2 that the IA 64 and IA-32 nodes are both are located at the same site. (Set environment variables properly)
15. Write a MPI program using two MPI attributes (MPICHX_TOPOLOGY_DEPTHS and MPICHX_TOPOLOGY_COLORS) associated with every MPI communicator.
16. Write RSL file and execute MPI gather collective communication library call program on set of '*p*' processes used in an iterative fashion. The MPICH-G2 strategy of ordering various communication methods based on performance asserting should be considered. The program is considered as a single job across *p* sites (each processes at each site). The aim is to measure the communication time for all iterations with input message size starting with message size 10 MB on each process, and maximum message size upto to 200 MB on *p* processes. In the first iteration, process with rank 0 acts as root, with message size of 10 MB. In the second iteration, root process is chosen among '*p*-1' process and MPI gather operation is performed with an increment of 10 MB on each process. The iterative process is continues till each process acts as root rank in *gather* communication. RSL file generated for each iteration
17. Write RSL file and two MPI programs based on MPI scatter collective communication and MPI broadcast library calls on set of '*p*' processes in an iterative fashion. The program is considered as a single job across *p* sites (each processes at each site). The MPICH-G2 strategy of ordering various communication methods based on performance asserting should be considered. The aim is to measure the communication time for the message size 10 MB to the maximum message size upto to 200 MB. In the first iteration, process with rank 0 acts as root process with a message size of 10 MB on root process. In the second iteration, root process is chosen among '*p-1*' process and MPI Scatter /Broadcast operation is performed with an increment of 10 MB on root processes. The iterative process is continues till each process acts as root rank in *scatter* /*broadcast* communication. RSL file generated for each iteration

Part-IX: Enabling Applications Enabling applications using Globus 4.x & Component based technology web services (XML, WSDL) and Grid Services)

Module Names : Module 24(GridProgApp-8)
Project Type : Short /Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Installation & Setup of Globus4.x; Globus4.x Components- GRAM, GridFTP, GSI, GASS, globus_io, Programs using XML-based Web Services for distributed computing approaches

Module: 24 Calendar Days 14

1. Develop and deploy examples using XML-based Web Services for distributed computing approaches.
2. Understand the concepts of web services (XML, UDDI, SOAP, WSDL) and grid services (Open Grid Services Architecture – OGSA)
3. Studies on integration of Grid and web services concepts and technologies will be done.
4. Study the basic concepts of common component architecture (CCA) for use in development of large-scale scientific and engineering applications
5. Demonstrate example using XML document and languages such as Java, C++, which describe the components of CCA.
6. Write a workflow of Grid Application using the XML in grid application and script language is used to mange the workflow of the Grid Application.

Part-IX: Enabling Applications for Grid Computing using Globus and Data Management components of Globus Toolkit

Module Names : Module 25(GridDataManagmt-1)
Project Type : Short /Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Installation & Setup of Globus4.0; Globus4.0 Components- GRAM, GridFTP, GSI, GASS, globus_io, Programs using two major components for data management in the Globus Toolkit – Data Transfer and access; Data

Module: 25 — Calendar Days 14

Example programs using two major components for data management in the Globus Toolkit (Data Transfer and access; Data replication and management)

1. Staging the files out with RSL (The output file is retrieved from the execution node and saved on the directory of the machine where the globusrun command was issued)
2. Stage the files on GridFTP Server (The output file is put on a GridFTP server running on site '*p1*')
3. Staging the files in: (Move the data from one site (location) to the other site (execution node) and the file is saved on user's home directory of client.)
4. Staging the files in: (Move the data from one site (location) to the other site (execution node) and the file is saved on user's home directory of client. Use the local GASS server started by globusrun)
5. Using file_stage_in_shared (Copy the file in the GASS cache directory)
6. Using file_stage_in – Example of failure; if the file is already there, the job will fail.
7. globus_io: Activating GSSAPI security on a socket communication
8. globus_io_attr_set_secure_authoriation_mode() – Determine whether to call the GSSAPI security context establishment functions once a socket connection is established (Authenticated connection)
9. Example on globus_io_attr_set_set_secure channel_mode() – to determine if any data wrapping should be done on the socket connection. Data protection is provided using GLOBUS_IO_SECURE_CHANNEL_MODE_GSI_WRAP with support of GSI features, such as delegation
10. Example on globus_io_attr_set_set_secure channel_mode() – to determine if any data wrapping should be done on the socket connection. Use GLOBUS_IO_SECURE_PROTECTION_MODE_PRIVATE for encrypted messages
11. Example on globus_to_attr_set_secure_delegation_mode() – used to determine whether the process credentials should be delegated to the other side of the connection. Use GLOBUS_IO_SECURE_DELEGATION_MODE_FULL_PROXY delegates full credentials to the server.
12. Example on globus_io_set_secure_proxy_mode() – to determine whether the process should accept limited proxy certificates for authentication. Use GLOBUS_IO_SECURE_DELEGATION_MODE_MANY to accept any proxy as a valid authentication

Part-IX: Enabling Applications for Grid Computing using Globus and Data Management components of Globus Toolkit

Module Names : Module 26 (GridDataManagmt -2)
Project Type : Short /Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Installation & Setup of Globus4.0; Globus4.0 Components- GRAM, GridFTP, GSI, GASS, globus_io, Programs using two major components for data management in the Globus Toolkit – Data Transfer and access; Data replication and management

Module: 26 Calendar Days 14

Write programs using two major components for data management in the Globus Toolkit (Data Transfer and access; Data replication and management)

13. Example on globus_io example: Establish a secure and authenticated communication between two hosts by using the globus_io functions
14. GASS Copy example: Copy a local file remotely via a GASS severs.
15. GASS server example: Use the globus_gass_erver_ez_API, and implement a simple GASS server.
16. GASS server example: Starting a GASS server
17. Globus cache management (Adding a file to the cache, Retrieving a file in the cache, Invoking a program from the cache)
18. Globus GridFTP APIs: Simple Authenticated Put – (transfer runs using all defaults)
19. Globus GridFTP APIs: Parallel transfer example (Parallelism – runs in Extended block mode which is a file transfer mode where data can be sent over multiple parallel connections and to multiple data storage nodes)

Part-X: Grid *GARUDA* Integration Test Scripts (GrINTeS v1.0)

Module Names : Module 27(GridTest-1); Module 28(GridTest-2);
Project Type : Short/Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand ; Globus4.0 Components- GRAM, GridFTP, GSI, GASS, Develop test suites and package GARUDA integration test script.

Module: 27 -28 **Calendar Days 14**

Development of programs for GARAUDA Integration test scripts using Globus Components

Interoperability Tests

1. Globusjobrun test
2. A globus-based "*hello, world*" test
3. Hello.rsl test
4. To test fully qualified domain names (FQDN) using *globus_libc_gehostname library call*
5. RSL Script: Create a proxy
6. RSL Script: Stage Test
7. RSL-Shell Test
8. Check for valid grid-proxy Test
9. GRAM Test
10. GRAM : BatchJob-submitt, and BatchJob-Query
11. GRAM : BatchJob-Cancel Tests, and BatchJob-Retrive Test
12. GRAM: Check for resources on Grid (use RSL String)
13. GASS Test,
14. GridFTP Test
15. GridFTP: Perform a third party file transfer Test
16. GSI : gsissh and gsiscp test
17. GRIS Test and
18. MDS Default Test
19. LDAP connection Test
20. Basic authentication Test
21. Query MDS test (local GIIS Test, JobManagers Test, Comparion Test)
22. Perl script to test TCP connection establishment
23. GARUDA globus-submission Test
24. GARUDA RSL Test
25. GARUDA GRAM Test
26. GARUDA MPI Test

Part- XI: Grid Probes (Perl & Globus : SPAGOM v1.0)

Module Names : Module 29(Gridprobes-1), Module 30 (GridProbes-2),
Project Type : Short /Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Installation & Setup of Globus4.0; Globus4.0 Components- GRAM, GridFTP, GSI, GASS, Deliver lectures, Demonstration, Execution, Performance of Probes

Module: 29-30 **Calendar Days 14**

Grid Probes (SPAGOM v1.0):

1. Probe Parameters: Check for valid grid-proxy,
2. Probe Parameters: Basic authentication for all grid nodes,
3. Probe Parameters: Query MDS
4. Ping Probe
5. Remote Ping Benchmark and stage back the results to Client
6. Ping-Pong Probe
7. Generalized Ping-Pong Probe
8. Sleep Benchmarks
9. Data Transfer Test
10. Circular probe
11. Gather Probe
12. Compute and Communication Benchmarks

Part-XII: Grid Application Benchmarks

Module Names : Module 31(GridApplnBench-1), Module 32 (GridApplnBench-2),
Project Type : Short/Long term
Time Duration : 14 Calendar Days (Overlap time schedule with other modules)
Nature of Work : Understand Installation & Setup of Globus4.0; Globus4.0 Components- GRAM, GridFTP, GSI, GASS, Deliver lectures, Execute of Benchmarks, Demonstrate Benchmarks, Performance of Benchmarks

Module: 31-32 **Calendar Days 14**

13. GRASP Benchmarks (Grid Assessment Probes)
14. GridBench Suite and Grid Bench Definition Language.
15. NAS Grid Benchmarks;
16. CrossGrid Application Benchmarks
17. Grid enabled LMBench
18. Micro, Macro and Application (NAS) Benchmarks
19. Numerical Libraries And The Grid; The Grads: Experiments with ScaLPACK
20. Benchmark Metrics & Performance Evaluation of Benchmarks

References:

1. *The Java CoG Kit User Manaul,* Gregor von Laszewski, Beulah Alunkal, Kaitzar Amin, Jarek Gawor, Mihael Hategan, San deep Nijsure. Available at: `http://www.globus.org/cog/manual-user.pdf`
2. Network Security Essentials: Application and Standards. Stallings, W. (2000), Prentice-Hall Inc.
3. A standard for architecture description, by R. Youngs, D. Redmond-Pyle, P. Spaas and E. Kahan. IBM Systems Journal, Volume 38, Number 1, 1999 Enterprise Solutions Structure available at:
4. *http://www.research.ibm.com/journal/sj/381/youngs.html*
5. On Death, Taxes, and the Convergence of Peer-toPeer and Grid Computing, by Ian Foster, Adriana lamnitch*i:*
6. *http://people.cs.uchicago.edu/Nanda/papers/foster_grid_vs_p2p.pdf*
7. Foster, et ai, The Grid: Blueprint for a New Computing Infrastructure, Morgan Kaufmann, 1999, ISBN 1558604758
8. The Anatomy of the Grid: Enabling Scalable Virtual Organizations, found at: *http://www.globus.org/research/papers/anatomy.pdf*
9. Tivoli License Manager *http://www.ibm.com/software/tivoli/products/license-mgr/*
10. IBM License Use Management *http://www.ibm.com/software/is/lum/*
11. Platofrm Global License Broker *http://www.platform.com/products/wm/glb/index.asp*
12. Globus Commodity Grid toolkts (CoGs) *http://www.globus.org/cog/*
13. Grid Computing Environment (GCE) *http://www.globus.org/research/development-environments.html*
14. Grid Enabled MPI *http://www.niu.edu/mpi/*
15. Grid Application Development software (GrADS) *http://nhse2.cs.rise.edu/grads/*
16. IBM Grid Toolbox `http`*://www.alphaworks.ibm.com/tech/gridtoolbox*
17. Grid Application Framework for Java *http://www.alphaworks.ibm.com/tech/GAF4J*
18. IBM Data Management products *http://www.ibm.com/software/data/*
19. Database Access and Integration Services Working group *http://www.gridforum.org/6_DATA/dais.htm*
20. MDS information provider example *http://www-unix.mcs.anl.gov/_slang/mds_iprovider_example/*
21. OpenSSH *http://www.openssh.org*
22. NFS Version 4 Open Source Reference Implementation; *http://www.citi.umich.edu/projects/nfsv4*
23. Avaki *http://www.avaki.com*
24. Global File System *http://www.sistina.com/products_gfs.htm*
25. Replica Location *Servicehttp://www.globus.org/rls*
26. Replica Location *Servicehttp://www.globus.org/rls*
27. GDMP *http://project-gdmp.web.cern.ch/project-gdl*
28. OGSA-D*AIhttp://umbriel.dcs.gla.ac.uk/NeSC/general/projects/OGSA_DAI/*
29. SOAP *http://www.cs.fsu.edu/Nengelen/soap.html*
30. Spitfire project *http://spitfire.web.cern.ch/Spitfire/*
31. Storage Resource Broker *http://www.npaci.edu/DICE/SRB/*
32. European Union data grid *http://www.eu-datagrid.org*
33. Griphyn Project *http://www.griphyn.org/*
34. Particle Physics Data Grid *http://www.ppdg.net/*
35. Grid Portal Development Kit *http://doesciencegrid.org/projects/GPDK/*
36. GSI-OpenSSH http://www.nsf-middleware.org/NMIR2/
37. Grid Portal Development Kit *http://doesciencegrid.org/projects/GPDK/*
38. CommonC++ *http://www.gnu.org/software/commonc++/*

39. *SOAP Plugin http://gsoap2.sourceforge.net*

40. Platform Computing *http://www.platform.com*

41. *DataSynapse http://www.datasynapse.com*

42. *United Devices http://www.ud.com*

43. Allan Snavely, Greg Chun, Henri Casanova, Rob Van der Wijgaart, Michael A Frumkin, Benchmarks for Grid Computing - A Review of Onging Efforts and Future Directions, Univ of California San Diego, NSF project, Copyright 2000 ACM, 2000

44. Data Grid, *http://eu-datagrid.web.cern.ch/eu-datagrid/*

45. Dongarra, J., Bunch, J., Moler C., an Stewart, G. W., LINPACK user guide, Tech report, 1979, *http://www.top500.org/list/linpack.php*

46. Evaluating the Information Power Grid using the NAS Grid Benchmarks, *http://www.nas.nasa.gov/Research/Reports/Techreports/2004/PDF/nas-04-005.pdf, 2004*

47. Foster, I. And C. Kesselman, The Grid: Blueprint for a New Computing Infrastructure, San Fransisco: Norgan Kaufmann Publishers, Inc., 1999.

48. Foster, I., C. Kesselman, and S. Tuecke, The Anatomy of the Grid: Enabling Scalable Virtual Organizations, International Journal of High Performance Computing Applications, Vol. 15, No. 3, 2001

49. Grid Computing, www.gridcomputing.com

50. Grid Assessment Probe benchmarks (GRASP), http://grail.sdsc.edu/projects/grasp/

51. GridBench, http://grid.ucy.ac.cy/Papers/gridb.pdf

52. Globus website http://www.globus.org/

53. GGF (Global Grid Forum) – http://www.gridforum.org/

54. Greg Chun, Holly Dail, Henri Casanova, Allan Snavely, Benchmark Probes for Grid Assessment Department of Computer Science and Engineering, University of California San Diego, NSF project

55. HPL benchmark, www.netlib.org/benchmark/hpl/

56. Hey, T., and Lancaster, D., The development of Parkbench and performance prediction, The Inter. J. of High Performance Computing Applications 14, No. 3, pp. 205-215, 2000

57. LMBench, http://www.bitmover.com/lmbench/

58. MPBench, http://icl.cs.utk.edu/projects/llcbench/mpbench.html

59. NAS Grid Benchmarks, http://www.nas.nasa.gov/Software/NPB

60. NASA's Information Power Grid (IPG), http://www.ipg.nasa.gov

61. STREAM: Measuring Sustainable Memory Bandwidth in High Performance Computing, http://www.cs.virginia.edu/stream.

62. Srikumar Venugopal, Rajkumar Buyya and Lyle Winton, A Grid Service Broker for Scheduling Distributed Data-Oriented Applications on Global Grids, 2004

63. The PALLAS MPI Benchmarks, http://www.pallas.com/

64. The P-COMS MPI Benchmarks: PARAM Padma , http://www.cdacindia.com/

65. The SPEC Benchmarks http://www.specbench.org

66. Zs. Németh: Invited talk, Grid performance, grid benchmarks, grid metrics, Proceedings of the 3rd Cracow Grid Workshop, Cracow, October, 2003. pp 34-41, http://www.lpds.sztaki.hu/~zsnemeth/

67. Catlett C., TeraGrid Primer, http://www.teragrid.org/about.html

68. IBM – Enabling Applications for Grid Computing with Globus International Technical Support Organization, http://ibm.com/redbooks, June 2003

69. Anotonie Pettitet, Susan Blackford, Jack Dongarra, Brett Ellis, Graham Flagg, Kenneth Roche, and Sathish Vadhiyar, Numerical Libraries And The Grid: The GrADS Experimnets *with ScaLAPACK,* UT-CS-01, 460 report, University of Tennesse, Knoxville TN 37996, April **(2001)**

70. Gropp, W. et al., A high Performance, Portable Implementation of the MPI message Passing Interface

Standard Parallel Computing, vol.22 (6); 00789 –828, (1996)

71. J. Dongarra, Top-500 web-site :www.top500.org

72. Whaley R., A Petitet, and J. Dongarra, Automated Empirical Optimization of Software and the ATLAS, Parallel Computing, Vol, 27 (1-2), page 03-25, (2001)

73. Bailey, D.H., T. Harris, W. Saphir, R. Van Der Wijingart, A. Woo, and M. Yarrow, The NAS Parallel Benchmarks 2.0, NASA Ames Research Center Report NAS-95-020 (1995)

74. PALLAS, Version 2.2.1, http://www.pallas.com/, (2002)

75. http://hpcf.nersc.gov/vendor_docs/ibm/essl/essl02.html,ESSL – Engg. And Scientific Subroutine Library (2004)

76. http://hpcf.nersc.gov/vendor_docs/ibm/pessl/pessl02.html, PESSL – Parallel Engg. And Scientific Subroutine Library, (2004)

77. http://www.nag.co.uk/numeric/numerical_libraries.asp; NAG– The Numerical Algorithms Group Ltd., (2004)

78. http://www-fp.mcs.anl.gov/petsc/, PETSc – Portable, Extensible Toolkit for Scientific Computation, (2004)

79. Karypis George and Vipin Kumar: METIS and ParMETIS, Serial and Parallel Software Packages for Partitioning Unstructured Graphs for Computing Fill-reducing Orderings of Sparse Matrices. Army High Performance Computing Research Center, University of Minnesota, http://www.cs.umn.edu/~karypis/metis/metis.html (1995)

80. Johnson A.A and T. E. Tezduyar, Parallel Computation of Incompressible Flows with Complex Geometries. International Journal for Numerical Methods in Fluids, Vol. 24: pp. 1321-1340 (1997)

81. P.Henon and Yousef Saad, A parallel multilevel {ILU} factorization based on a hierarchical graph decomposition, Report umsi-2004-xx1, Minnesota Supercomputer Institute, University of Minnesota, Minneapolis, MN, (2004)

82. Yousef Saad and Masha Sosonkina, pARMS : A package for the parallel iterative solution of general large sparse linear systems user's guide, Report UMSI2004-8, Minnesota Supercomputer Institute, Univ of Minnesota, Minneapolis, MN (2004).

83. Foster, I., C. Kesselman, and S. Tuecke, The Anatomy of the Grid: Enabling Scalable Virtual Organizations, International Journal of High Perf. Computing Applications, Vol. 15, No. 3, (2001)

84. Indrajit S. Dhillon, Narendra K. Karmarkar, K.G. Ramakrishnan, " An Overview Of The Compilation process For A New Parallel Architecture", Mathematical Sciences Research Center, AT&T Bell Laboratories, Indian Institute of Technology-Bombay, April, (1991)

85. Narendra K. Karmarkar "A New Parallel Architecture For Sparse Matrix Computations", Proc.Of The Work Shop On Parallel Computing, BARC, BOMBAY, 1-18, February (1990)

86. Thomas Sterling," Interdependence of architecture and software for effective terascale computing", PRIMEUR, The monthly news service for the European HPCN Community, Issue 27, June (2003)

87. Tom R Halfhill, "IBM Makes Designer Genes", Micro Processor, MicroProcessor Report, @IN-STAT-MDR, URL:http://researchweb.watson.ibm.com/bluegene/; October 11, (2004)

88. Exploring the benefits of FPGA-processor technology for genome analysis at Acconovis; ISC 2003 Conference; Heidelberg, URL : http://www.acconovis.com; PRIMEUR, The monthly news service for the European HPCN Community, Issue 27, June (2003)

89. High Performance Computing –FPGA technology; URL:http://www.cdac.in;

90. Ernst L. Leiss, Parallel and Vector Computing A practical Introduction, McGraw-Hill Series on Computer Engineering, Newyork (1995).

1. Peter S.Pacheco, Parallel Programming with MPI, Morgan Kaufmann Publishers,Inc San Francisco,California (**1997)**
2. Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar, *Introduction to Parallel Computing*, Second Edition, Harlow, England, Addison Wesley (**2003)**
3. William Gropp, Rusty Lusk, Tuning MPI Applications for Peak Performance, Pittsburgh (**1996)**

91. Ian T. Foster, Designing and Building Parallel Programs, Concepts and tools for Parallel Software Engineering, Addison-Wesley Publishing Company (1995)

92. Kai Hwang, Zhiwei Xu, Scalable Parallel Computing (Technology Architecture, Programming), McGraw Hill International Editions, (1998)

93. Kai Hwang and Fave A Briggs, Computer Architecture and Parallel Processing, McGRAW HILL International Editions (1984)

94. Culler David E, Jaswinder Pal Singh with Anoop Gupta, Parallel Computer Architecture, A Hardware/Software Approach, Morgan Kaufmann Publishers, Inc, Reprinted (2004)

95. Quinn Michael J., Parallel Programming in C with MPI and OpenMP, Tata-McGRaw-Hill Edition, (2003)

96. Rajeev Matwani and Prabhakar Raghavan , Randomised Algorithms, Combridge University Press, (2004)

97. Nancy A Lynch, Distributed Algorithms, Morgan KAYFMANN, (2003)

98. Sartaj Sahani, Data Structures Algorithms and Applications in JAVA, (2005)

99. William Gropp, Rajeev Thakur: Using MPI-2 Advanced Features of the Message-Passing Interface, The MIT Press, (2000)

100. The Message Passing Interface (MPI), Standard; http://www.unix.mcs.anl.gov/mpi/

101. NTTCP : New TTCP program, http://www.leo.org/~elmar/nttcp/

102. NetPIPE ; New TTCP Program, http://www.scl.ameslab.gov/netpipe/

103. LLCBench Home Page. http://icl.cs.utk.edu/projects/llcbench/

104. P-COMS http://www.cdac.in/

105. The High Performance Computing Center (Höchstleistungsrechenzentrum) Stuttgart (HLRS) of the University of Stuttgart; http://www.hlrs.de/organization/par/