Grid Programming with Globus APIs

Ravi T. & Shamjith K.V.
System Software Development Group,
CDAC, Bangalore.
Presentation Contents

- Globus Tool kit Services
- Globus Programming Conventions
- Overview of the main Globus APIs
- Case Study : Example Application
Globus Toolkit Services

Globus Toolkit provides a set of services that can be combined to build a wide variety of Grid applications and systems:

- Globus Security Infrastructure (GSI)
- Globus Resource Allocation Manager (GRAM)
- Grid File Transfer Protocol (GridFTP)
- Monitoring and Discovery Service (MDS)
Globus Programming Conventions
Naming Conventions

- **Globus SDK** is a set of packages
- **Package** is set of related modules
- **Globus builds** happen with “flavors”
- Command line tool names start with `globus` or `grid`.
  - Ex: `globus-job-run`
- Header files will be located in `$GLOBUS_LOCATION/include/<flavor>` directory.
- Libraries will be put in `$GLOBUS_LOCATION/lib` directory.
- Library names follow the same naming convention as the functions in that library: `libglobus_<package>_:<module>_<flavor>.a`
  - Examples: `libglobus_gram_client_gcc32dbg.a`, `libglobus_rsl_assist_gcc32dbg.a`
Function return values

- All function names and other exported symbols follow the form:
  
  `globus_<package>_<module>_<action>`
  
  Example: `globus_gram_client_job_request()`

- Most functions return value either `int` or `globus_result_t` (in `globus_result_t` (in new APIs)

- `GLOBUS_SUCCESS` returned if the operation was successful

- In case of error, `int` gives directly the error number.

- `globus_result_t` can be expanded to an "object" (error type hierarchy with single inheritance) with `globus_error_get`.

- `globus_error_print_friendly` and `globus_error_print_chain` can be used to get a user friendly error message.
Common Services

- Will come as `globus_common` library.
- Assist in writing portable Makefiles and programs.
- Fundamental modules are
  - Module activation/deactivation
  - Portable thread library (POSIX subset)
  - Thread-safe and portable libc wrappers
  - Timed and periodic callbacks
  - Modules to manipulate lists, fifos, URLs, ...
- The rest of the Globus Toolkit relies on `globus_common`
Writing Makefiles

• **globus-makefile-header** can be used to generate platform and installation specific information.

• **Input:** flavor and a set of globus modules.

• **Output:** variables to be included and used in make files.

• **E.g.:**

$$\texttt{GLOBUS\_LOCATION/bin/globus-makefile-header--flavor=gcc32dbg}$$

$$\texttt{globus\_io}$$
Thread Programming

- wrapper functions for threads, which call the underlying underlying thread implementation
globus_thread_*, globus_mutex_*, globus Cond_*,
globus_cond_*()
- Simple POSIX threads (pthreads) subset
- Same arguments and semantics as pthreads
- Simply change “pthread” to “globus” or “globus_thread” in the function name
- Co-exists with programs using pthreads directly directly
Activating / Deactivating the Modules

• All Globus modules must be initialized before use by calling the `globus_module_activate()` function
• Similarly, `globus_module_deactivate()` must be called to shutdown the module
• Support for simultaneous activations of a module
• Dependencies between modules will be handled internally.
• **typical code:**

```c
#include "globus_common.h"
#include "globus_io.h"

int main() {
    // ...
    globus_module_activate(GLOBUS_IO_MODULE);
    // ... (use)
    globus_module_deactivate(GLOBUS_IO_MODULE);
    // ...
}
```
**globus_libc**

- Wrappers around standard libc functions
  - Thread safe, even if underlying libc is not
  - Same interface for threaded and non-threaded as with normal libc
    - POSIX reentrant functions (*_r()) always work
    - Example: `globus_libc_gethostbyname_r()`
  - Fixes or enhances some functions
    - Example: `globus_libc_gethostname()` tries to figure out a fully qualified hostname despite system configuration
Convenience Modules

- **globus_common** also contains a small set of convenience convenience modules
  - **globus_fifo**: First-In-First-Out queue
  - **globus_hashtable**: Hash table
  - **globus_list**: List functions
  - **globus_symboltable**: Symbol table management
  - **globus_url**: URL parsing
  - **globus_error**: Error retrieving.
Overview of the Main Globus APIs
Security Modules

- Implemented by the `gss` module
- `globus_gss_assist` module provides an easier interface
- Possible to attach to GSS-API directly, but other Globus Globus modules already use it internally
  Examples: Globus-I/O for socket connections, GASS and GridFTP for file transfers
- The `globus_gsi_proxy` library is for the proxy creation and delegation process.
- The `globus_gsi_credential` library for handling X.509 based X.509 based PKI credentials
Resource Management

- The (GRAM) protocol and its client API allows programs to be started on remote resources, despite local heterogeneity.
- RSL is used to communicate requirements.
- A layered architecture allows application-specific resource brokers and co-allocators to be defined in terms of GRAM services.
  - Integrated with Condor, PBS, MPICH-G2, …
Resource Management Architecture

Application → RSL → Broker → RSL specialization → Information Service

Ground RSL → Co-allocator → Queries & Info

Local resource managers:
- GRAM
- LSF
- Condor
- GRAM
- NQE

Simple ground RSL
Resource Specification Language

- Notation for exchange of information between components
  - Syntax similar to MDS/LDAP filters
- RSL can provide
  - **Resource requirements**: Machine type, number of nodes, memory, etc.
  - **Job configuration**: Directory, executable, args, environment
- GTK provides an API/SDK for manipulating RSLs.
Resource Management APIs

- The globus_gram_client API provides access to all of the core job submission and management capabilities, including callback capabilities for monitoring job status.
- The globus_rsl API provides convenience functions for manipulating and constructing RSL strings.
- The globus_gram_myjob allows multi-process jobs to self-organize and to communicate with each other.
- The globus_duroc_control and globus_duroc_runtime APIs provide access to multirequest (co-allocation) capabilities.
Communication Modules

- **globus_io**: provides wrappers for using TCP and UDP sockets and file I/O
  - Same function as in calls in Unix and Win32
  - Easy to add authentication and encryption
- **mp**: a simple wrapper around MPI (message passing libraries)
- **nexus**: a multi method communication library used in Globus
  - Globus 1.X, now deprecated
Data Management

Metadata Catalog

Application

Replica Catalog

Replica Selection

Replica Location 1
Replica Location 2
Replica Location 3

Attribute Specification

Logical Collection and Logical File Name

Selected Replica

Multiple Locations

Performance Information & Predictions

GridFTP Control Channel

Tape Library

Disk Cache

Disk Array

GridFTP Data Channel
Grid FTP Modules

- **globus_ftp_client**
  - Provides FTP-like commands (get, put, etc.)
  - Plug-in architecture to provide access to interesting events
  - Plug-in architecture for installing application or grid-specific fault recovery and performance tuning algorithms within the library

- **globus_ftp_control**
  - Lower level API for detailed control
Access/Transport Protocol Requirements for FTP

- Requires below suite of communication libraries and related tools that support
  - GSI, Kerberos security
  - Third-party transfers
  - Parameter set/negotiate
  - Partial file access
  - Reliability/restart
  - Large file support
  - Data channel reuse
  - Integrated instrumentation
  - Logging/audit trail
  - Parallel transfers
  - Striping
  - Policy-based access control
  - Server-side computation
  - Proxies (firewall, load bal)

- All based on a standard, widely deployed protocol called ‘GridFTP’
Grid FTP – FTP Operations Supported

- File / Directory Existence
- Make Directory
- Remove Directory
- Delete
- List
- MList
- Move
- Chmod

- Get
- Put
- Modification Time
- Size
- Chksum
- Abort
A Word on GASS
(Globus Access to Secondary Storage)

• GASS provides services for file and executable staging and I/O and I/O redirection that work well with GRAM.

• GASS uses GSI-enabled HTTP as the protocol for data transfer, transfer, and a caching algorithm for copying data when necessary.

• `globus_gass`, `globus_gass_transfer` and `globus_gass_cache` APIs provide programmer access to these capabilities, which are already integrated with the GRAM the GRAM job submission tools.
Monitoring and Discovery Service (MDS)

- Globus uses the LDAP based GRIS/GIIS servers to store information about resources.
- This can be used to store additional "telephone directory" type data.
- RFC 1823 defines a draft standard C client API to access LDAP databases, Globus uses the OpenLDAP client library.
Two classes of MDS servers

- Grid Resource Information Service (GRIS)
- Grid Information Index Service (GIIS)
MDS Approach

- Based on LDAP
  - Lightweight Directory Access Protocol v3 (LDAPv3)
- Globus Toolkit schema
  - Host-centric representation
MDS Components

- **LDAP 3.0 Protocol Engine**
  - Based on OpenLDAP
  - Integrated caching

- **Information providers**
  - Delivers resource information to backend

- **APIs for accessing & updating MDS contents**
  - C, Java, PERL (LDAP API, JNDI)
MDS Commands

- LDAP defines a set of standard commands
  ldapsearch, etc.
- MDS-specific commands
  - grid-info-search, grid-info-host-search
- APIs are defined for C, Java, etc.
  - C: OpenLDAP client API
    - ldap_search_s(), …
  - Java: JNDI
Case Study
Case Study: Building a Sample Application

Pi (π) Calculation using the Monte Carlo method

Globus APIs used for:
- Authentication
- Job Submission
- Co-allocation
- Data Movement
Thank you!