GWD-I
Category: Informational
Judith Utley, Editor
NASA Ames Research Center
March 2004

PGM-RG
June 2003

Workshop on Tools For Grid Management
Held in conjunction with GGF-8, June 26, 2003, Seattle, WA, USA

Preceedings

Status of This Memo

This memo provides information to the Grid user community. It does not define any standards or technical recommendations. Distribution is unlimited.

Abstract

This is the proceedings of the Workshop on Tools For Grid Management that has been organized by the Production Grid Management Research Group (PGM-RG). It contains the abstracts of the talks accepted by the program committee and the presentations given at the workshop.

Contents

Foreward

The Production Grid Management Research Group (PGM-RG) brings together grid practitioners to discuss issues encountered in managing a grid and the hurdles, both technical and non-technical, to overcome in moving a grid to the persistent or production stage. The group also explores new paradigms in supporting and managing grids.

The goals of this workshop were to explore tools, both available and under development, for managing a grid.

Suggested topics for talks were

- certificate management,
- grid status monitoring
- user support tools
- software installation and configuration
- account management
- environment management

Due to the short time available to announce the workshop the presenters were chosen from submitted abstracts as well as invited speakers.

Security Considerations

There may be security issues related to the individual tools presented at the workshop. These need to be considered on a per-tool basis.

Author Information

The workshop was organized by the chair persons of the PGM-RG:

- Judith Utley, NASA Ames Research Center, USA, utley@nas.nasa,.gov
- Doru Marcusiu, National Center Supercomputing Application, marcusiu@ncsa.uiuc.edu
- Franz-Joseph Pfreundt, ITWM, pfreundt@itwm.fhg.de

The authors of the individual presentations can be reached according to the information provided in the respective presentations.

Intellectual Property Statement

The GGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the GGF Secretariat.

The GGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the GGF Executive Director.

Full Copyright Notice

Program Committee

The co-chairs of the PGM-RG served as the program committee for this workshop:

- Judith Utley, NASA Ames Research Center, USA, utley@nas.nasa,.gov
- Doru Marcusiu, National Center Supercomputing Application, marcusiu@ncsa.uiuc.edu
- Franz-Joseph Pfreundt, ITWM, pfreundt@itwm.fhg.de

Workshop Presentations

- *GSI Credentail Management with MyProxy*, Jim Basney, National Center for Superomputing Applications (NCSA)

- *A System for Monitoring and Management of Computational Grid,* Warren Smith, CSC/NASA Ames Research Center

- *GRIP: Creating Interoperability between Grids,* Philipp Wieder, Central Institue for Applied Mathematifs (ZAM), Research Center Julich

- *Grid Weaver,* Peter Toft, HP Labs

- *Automatically Establishing the Executive Environment for User Applications,* Paul Kolano, AMTI/NASA Ames Research Center

- *Exegrid – Workflow Support and Resource Management,* Jurgen Falkner, Fraunhofer Resource Grid, Fraunhofer IAO

- *Grid Packaging Using GPT 3.x,* Patrick Duda, National Center for Supercomputing Applications (NCSA)

GGF-8 Workshop on Tools For Grid Management

GSI Credential Management with MyProxy

Jim Basney, National Center for Supercomputinmg Applications (NCSA)

Abstract

The Grid Security Infrastructure (GSI) can be a source of confusion and frustration for Grid users, resulting in a support nightmare for Grid administrators. The process of initially obtaining a GSI credential is one of the user's first experiences on the Grid: it should be as straightforward as possible so users can get on the Grid quickly and easily as opposed to turning away in frustration. Once obtained, the user's credential must be both well-secured and conveniently accessible

This talk will present the credential management services provided by the MyProxy Online Credential Repository. Storing users' credentials in a MyProxy repository can ease credential distribution and management. Rather than requiring users to go through a complicated PKI enrollment process to obtain credentials, administrators can load user credentials into the MyProxy repository and simply distribute passwords to the users for retrieving their credentials from the repository. Rather than managing certificate and key files themselves, users can retrieve proxy credentials directly from the MyProxy repository when needed. Administrators can easily update the credentials in the repository to renew credentials or reset forgotten user passwords.

Grid administrators can use MyProxy to manage the security of credentials at their site. MyProxy can enforce password policies to ensure that credentials are encrypted with well-chosen passwords. Efforts for securing credentials and monitoring their use can be focused on the MyProxy repository, where the long-term credentials are stored, rather than having credentials distributed across many end-user systems out of the administrator's control. Users retrieve proxy credentials from MyProxy with short-lifetimes that limit risk of compromise.

The talk will describe the current version of the MyProxy software along with our plans for future developmen More information available at http://myproxy.ncsa.uiuc.edu/

Presentation

GSI Credential Management with MyProxy

GGF8 Production Grid Management RG Workshop
June 26, 2003

Jim Basney
jbasney@ncsa.uiuc.edu
http://myproxy.ncsa.uiuc.edu/

NCSA

MyProxy

- Online repository of encrypted GSI credentials
- Provides authenticated retrieval of proxy credentials over the network
- Improves usability
  - Retrieve proxy credentials when/where needed without managing private key and certificate files
- Improves security
  - Long-term credentials stored encrypted on a well-secured server

June 26, 2003 GSI Credential Management with MyProxy 2

NCSA

MyProxy Software

- Server and client tools available from http://myproxy.ncsa.uiuc.edu/
  - GPT packages for Globus Toolkit 2.2 & 2.4
  - Also included in NMI Release 3.0 at http://www.nsf-middleware.org/
- Compatible client implementations also available in Commodity Grid Kits
  - http://www.globus.org/cog/
- Supported by Grid Portal toolkits
  - Grid Portal Development Kit (GPDK): http://doesciencegrid.org/projects/GPDK/
  - Grid Portal Toolkit (GridPort): https://gridport.npaci.edu/
  - Xportlet: http://www.extreme.indiana.edu/xportlets/project/
- OGSI development in progress

Grid Security Infrastructure

- Credentials
  - Asymmetric public/private key pair
  - X.509 certificate, signed by Certificate Authority, binds identity to key pair
- Authentication (Who are you?)
  - Proof of possession of private key
  - Verify CA signature on X.509 certificate
- Authorization (What can you do?)
  - Based on certificate identity
  - Can be mapped to local Unix account

Credential Management

- Enrollment: Initially obtaining credentials
- Security: Protecting credentials (private keys)
- Accessibility: Getting credentials when needed
- Renewal: Handling credential expiration
- Translation: Using existing credentials to obtain credentials for a new mechanism or realm
- Delegation: Granting specific rights to others
- Control: Monitoring and auditing credential use
- Revocation: Handling credential compromise

Issuing Credentials via MyProxy

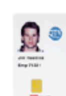
- Generate credentials on user's behalf and load into MyProxy repository
- Distribute MyProxy usernames and passphrases
  - Can use existing site usernames/passphrases
- Private key never leaves MyProxy repository
  - Proxy credentials delegated with configured max. lifetime
- Revoke credentials by removing from repository
- Provides a single point for focusing credential protection and usage monitoring
  - Enforce password policies
- Manage credentials on the user's behalf
  - Renew credentials before they expire
  - Reset forgotten credential passphrase

Integrating MyProxy with CA

- Using Globus SimpleCA
  – myproxy-admin-adduser generates SimpleCA credentials and loads them into repository
- Using existing CA
  – Create credentials as usual
  – Load with myproxy-admin-load-credential
- MyProxy need not be the only method of credential issuance
  – Can continue to issue credentials directly to experts to manage themselves

Alternatives: Smart Cards

- An excellent solution but costly
  – User-managed, portable credential storage
  – Security analogous to car keys or credit cards
    • Must be re-issued when lost or stolen
  – Private keys stay in hardware
  – Cards can be distributed with credentials pre-loaded
  – Card standards are mature
  – Costs are decreasing but still significant
    • $20 readers, $2 cards
    • Government ID card deployments
  – Some support already in GSI libraries
- MyProxy provides a "virtual smart card"
  – When smart card support is not ubiquitous or is too expensive



Alternative: Online CAs

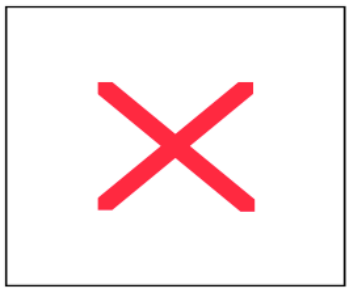
- A good solution with low administrative costs
  - User authenticates to online CA to obtain credentials immediately
    - No manual administrative approval required
  - Leverages existing authentication mechanisms (password, Kerberos, etc.)
  - Signs long-term or short-term credentials:
    - If long-term, then credentials are user-managed
    - If short-term, credentials retrieved on demand, without need for user key management
  - Examples: KCA and CACL
- MyProxy can be more flexible
  - Managing credentials from multiple CAs
  - In the future, managing multiple types of credentials

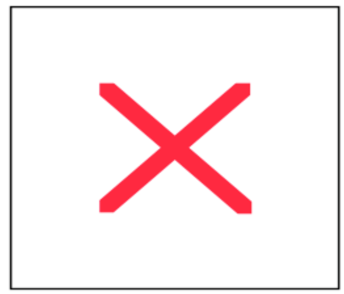
Credential Accessibility with MyProxy

- A MyProxy server can be deployed for a single user, a virtual organization, or a CA
- Users can delegate proxy credentials to the MyProxy server for storage
  - Can store multiple credentials with different names, lifetimes, and access policies
- Then, they can retrieve stored proxies when needed using MyProxy client tools
  - And allow trusted services to retrieve proxies
- No need to copy certificate and key files between machines

Delegation to Grid Portals

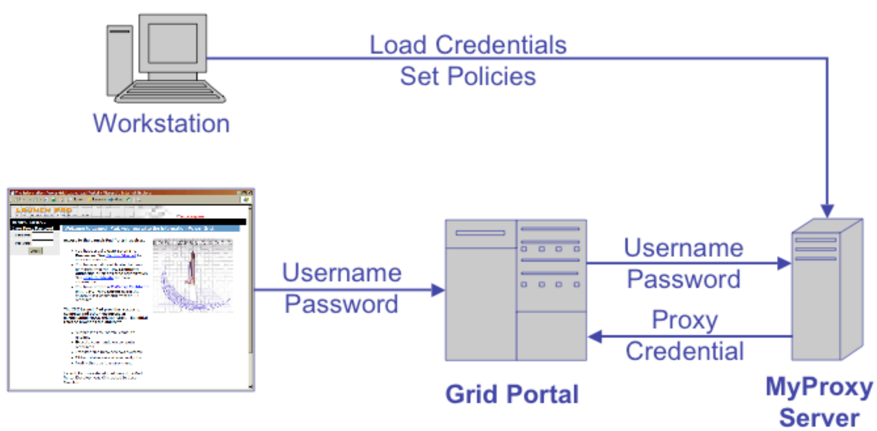
- Provide a web interface to Grid services
- Require credentials to act on user's behalf
- Use MyProxy to delegate credentials to portal





June 26, 2003 GSI Credential Management with MyProxy 11 NCSA

Delegation to Grid Portals

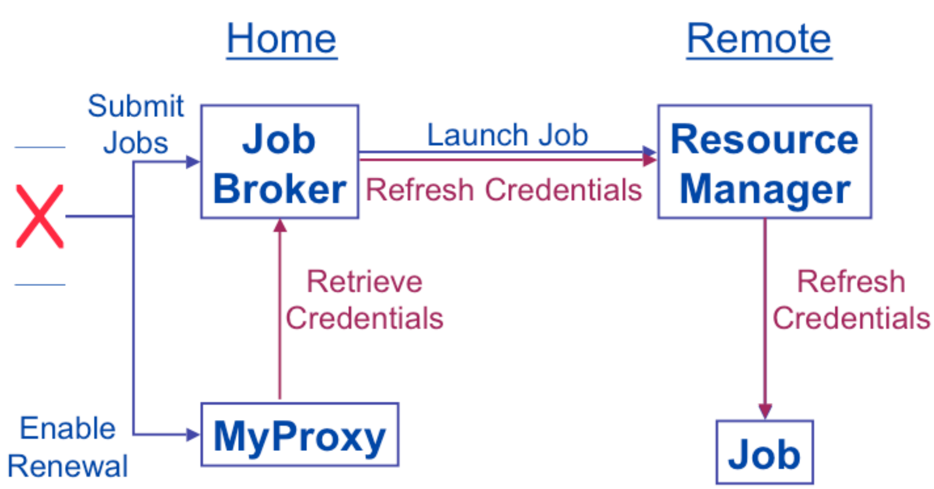


June 26, 2003 GSI Credential Management with MyProxy 12 NCSA

Credential Renewal

- Long-lived tasks or services need credentials
  - Task lifetime is difficult to predict
- Don’t want to delegate long-lived credentials
  - Fear of compromise
- Instead, renew credentials with MyProxy as needed during the task’s lifetime
  - Provides a single point of monitoring and control
  - Renewal policy can be modified at any time
    - For example, disable renewals if compromise is detected or suspected
- Integration with Condor-G in progress

Credential Renewal



MyProxy

- Provides a solution today for many GSI credential management issues
  - Enrollment
  - Private key security
  - Accessibility
  - Renewal
  - Passphrase-based delegation
  - Revocation and passphrase reset
- Work in progress
  - MyProxy OGSA Service
  - MyProxy Auditing
  - Credential Wallet for the Grid

MyProxy OGSA Service

- Credential manager factory
- Credential manager object leverages OGSI services
  - Query credential info via service data query
  - Remove credentials by destroying service instance
  - Monitor credential access via service notifications
  - Control credential access via standard service access control mechanisms
- Goal: A lightweight credential management service that can be easily instantiated when needed
- Good user interface is essential

MyProxy Auditing

- Develop standard OGSA audit service to which the MyProxy server logs activity
- Provide a secure query and notification interface
  - Credential owners can monitor use of their credentials and detect unauthorized use
  - Administrators can detect and investigate credential misuse

Credential Wallet for the Grid

- Provides an interface to my credentials
  - Multiple X.509 ID certificates, authorization credentials, CA certificates with CRLs
  - Supports multiple authentication mechanisms
  - Easily add, remove, modify credentials
  - Control credential access policies
  - Create authorization credentials for delegation
  - Receive event notifications
- Single sign-on unlocks wallet
  - Grid protocols negotiate for required credentials
  - Automatically retrieve needed credentials from wallet

Acknowledgements

- MyProxy Team (2002-2003)
  - NCSA: Shiva Shankar Chetan, Feng Qin, Zhenmin Li, Asita Anche, Vivek Sundaram, Praveen Appu
  - UVA: Marty Humphrey, Shaun Arnold, Dhiraj Parashar
  - Other authors/contributors: Jarek Gawor, Daniel Kouril, Jason Novotny, Miroslav Ruda, Benjamin Temko, Von Welch
- Financial Support

A System for Monitoring and Management of Computational Grids

Warren Smith, CSC/NASA Ames Research Center

Abstract

As organizations begin to deploy large computational grids, it has become apparent that systems for observation and control of the resources, services, and applications that make up such grids are needed. Administrators must observe resources and services to ensure that they are operating correctly and must control resources and services to ensure that their operation meets the needs of users. Users are also interested in the operation of resources and services so that they can choose the most appropriate ones to use. In this talk we describe a prototype system to monitor and manage computational grids and describe the general software framework for control and observation in distributed environments that it is based on.

Presentation

A System for Monitoring and Management of Computational Grids

Warren Smith
Computer Sciences Corporation
NASA Ames Research Center

Motivation

- Computational grids:
  - Many different types of resources
  - Services deployed on those resources
  - Applications executed by users
- There will be failures
  - Failures need to be observed
  - Observation of failures need to be communicated
- A grid must be managed
  - Failure management
  - General administration

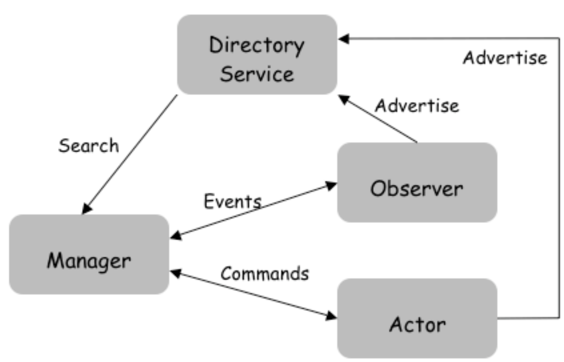
Approach

- Develop a general framework for observation and control
  - Observe and control a variety of resources and services
  - Operate in a distributed environment
  - Secure
  - Scalable
- Use this framework to monitor and manage grids
  - Observe computer systems, storage systems, networks
  - Observe job submission, information, file transfer services
  - Start, stop, and configure services
  - Notify administrators of problems
- Help develop and be compatible with standards
  - Global Grid Forum

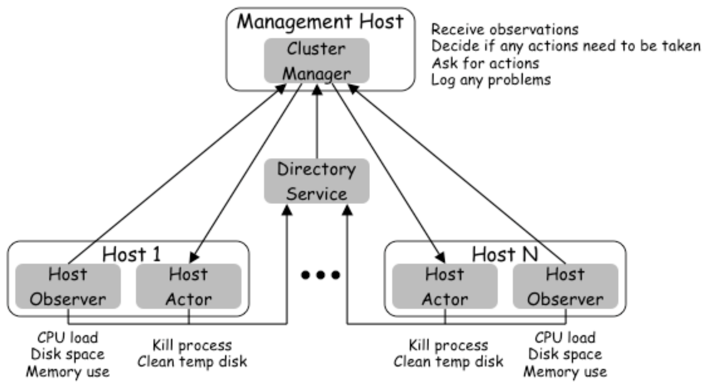
Why not use an existing system?

- Commercial systems
  - Many fully-featured tools available
  - Cost that could be too high for smaller partners
  - Incompatibility between different tools
  - Incompatible with grid security and authentication mechanisms
- Open source systems
  - Not as many features
  - Incompatible with each other
  - Not compatible with grid security mechanisms
- Either
  - Want a testbed for standardization

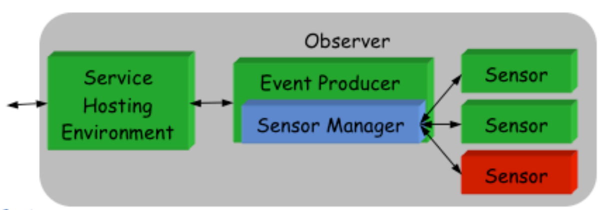
High-Level Architecture



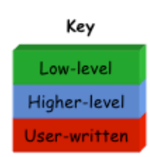
Monitoring and Managing a Cluster



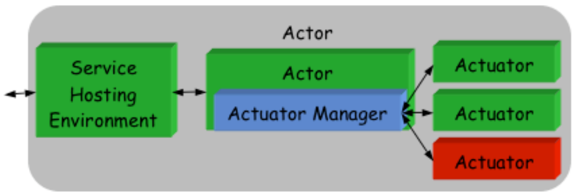
Observer



- Sensor
  - Performs a measurement and reports results
- Sensor manager
  - Manages sensors, subscriptions, and queries
- Event Producer
  - Subscribe
  - Query
  - Available events
  - Event schemas
- Service Hosting Environment



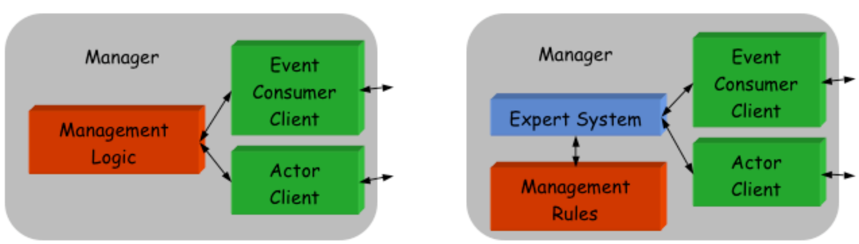
Actor



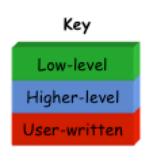
- Actuator
  - Performs an action
- Actuator Manager
  - Handles requests for actions by calling actuators
- Actor
  - Request action (RPC)
  - Available actions
  - Action schemas
- Service Hosting Environment

GGF8 PGM Workshop 8

Manager



- Two external interface components
  - Event Consumer Client
  - Actor Client
- 2 approaches to higher-level components
  - User writes management logic
  - User writes management rules and uses an expert system



GGF8 PGM Workshop 9

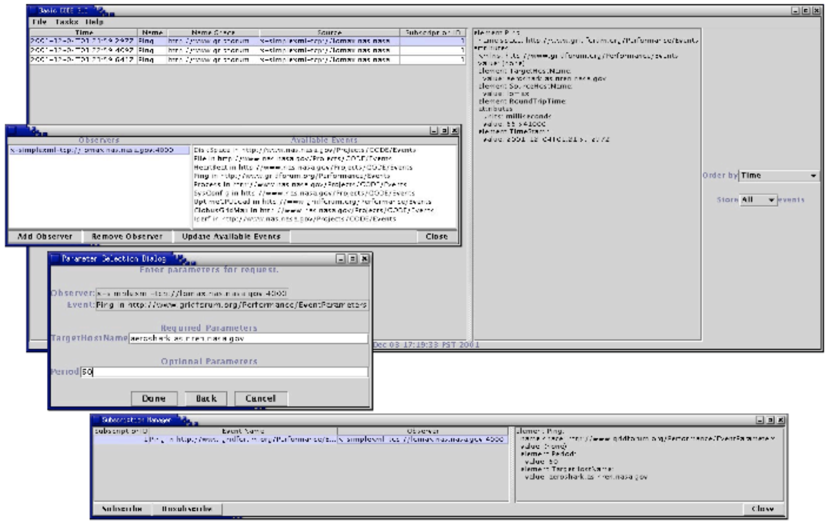
Directory Service

- Information about observers and actors
  - Contact location and protocol
  - Available events and actions
  - Who has access
- Dictionary
  - Event and action schemas
- Future: Information about event consumers
  - Archives
  - Channels
- Experimental component

Security

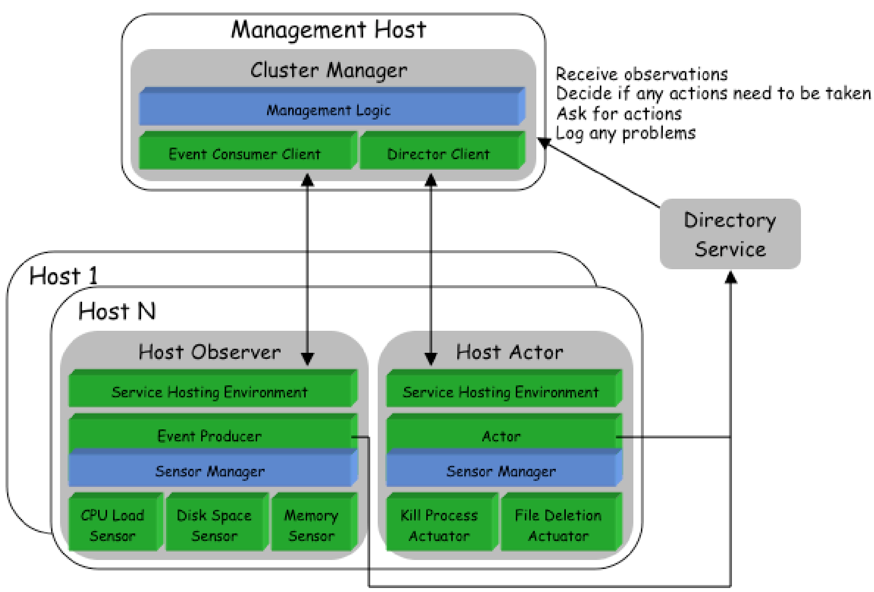
- GSI security
- Encrypted communication
  - SSL/TLS
- Authentication
  - X.509 certificates
  - Proxy certificates
- Authorization
  - Per-observer and per-actor
  - Pluggable user-defined authorization module
    - Module for X.509 subject-based access control lists available
  - Future: per-sensor and per-actuator

Basic GUI



GGF8 PGM Workshop 12

Monitoring and Managing a Cluster



GGF8 PGM Workshop 13

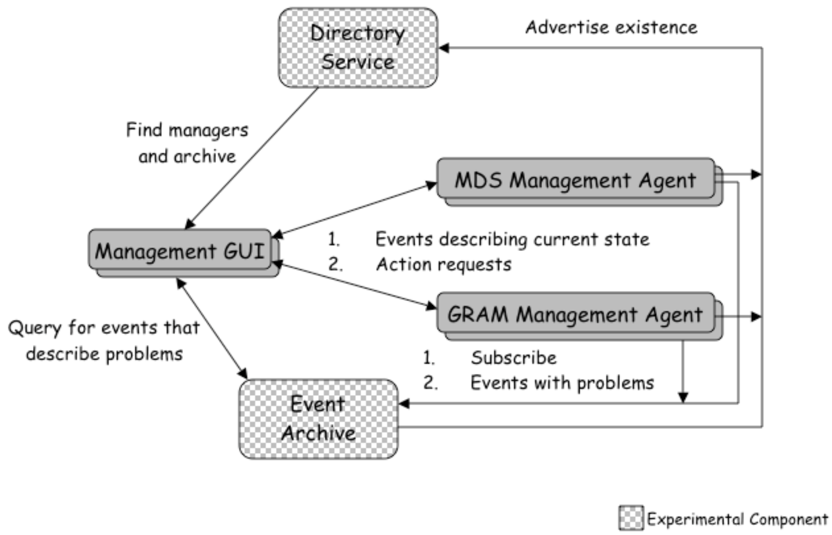
Implementation

- Communicates using TCP, UDP, or SSL
- XML encoding of messages
- C++ version
  - pthreads
  - Xerces XML parser
  - Globus I/O for authenticated and secure communication
  - Currently runs under IRIX, Solaris, Linux
  - CLIPS expert system
- Java version
  - Xerces XML parser
  - Globus Java CoG for authenticated and secure communication
  - JDK 1.3.x or 1.4.x

Grid Management System

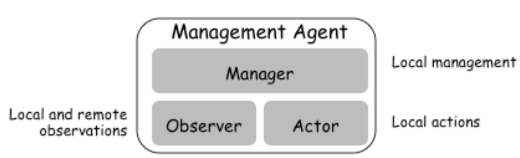
- Things to observe:
  - Resource status and usage
    - Computer systems and networks
  - Grid services
    - GRAM, MDS
    - Includes processes, log files, and test queries
- Things to control:
  - Add/remove user mappings in grid-mapfiles
  - Starting and stopping MDS servers
  - Add/remove/update CA certificates
- Provide a nice GUI to do all this

Grid Management System



Management Agent

- Management agents:
  - Perform observations
  - Perform actions
  - Manage local problems
    - Not doing any management right now
- Handle local problems locally



GRAM Management Agent

- Observes:
  - Network latency between GRAM hosts: ping
  - Available network bandwidth between hosts: IPerf
  - CPU load: Unix uptime, PBS qstat, LSF bjobs
  - Available memory: vmstat?
  - Available disk space: df
  - The Globus GRAM service: Log files
- Performs actions:
  - Modify Globus grid-mapfile
  - Start/stop IPerf server
  - Send email
- In the future will manage local problems
  - Receive local observations
  - Perform local actions when necessary

GGF8 PGM Workshop 18

MDS Management Agent

- Observes:
  - Network connectivity between GIS hosts: ping
  - CPU load: uptime
  - Available memory: vmstat?
  - Available disk space: df
  - The status of the LDAP server
    - The LDAP server process: ps
    - If LDAP queries are successful: ldap_search()
- Performs actions:
  - Start and stop LDAP server
  - Send email
- In the future will manage local problems

GGF8 PGM Workshop 19

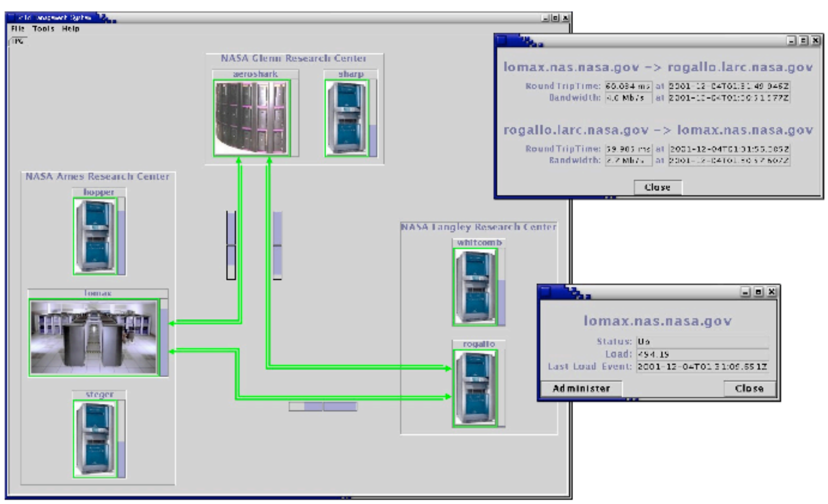
Ames Research Center

Event Archive

- Allows events to be archived and searched
- An XML database
  - Currently Xindice
  - Compatible with our XML-based events
- Queried using the Xpath language
- Use for all events, just errors, …
- Experimental component

Ames Research Center

Grid Management GUI



Grid Management GUI

- Similar to many you've seen before
- Java program
- Load on systems
  - System up or down
- Latency and bandwidth of network
  - Network up or down
- XML configuration file defines GUI
  - Which systems to monitor
  - Which sensors to use on each system
  - Where to place information on the screen
- More detailed information available as dialogs

Standardization

- Performance Working Group of the Grid Forum
  - Architecture
  - Event representations
  - Directory service schema
  - Producer-consumer communication protocols
- Grid Monitoring Architecture Working Group
- DAMED Working Group
- Grid Event Service Working Group?
  - BOF at next GGF, hopefully

Status and Future Work

- Current Status:
  - Worldwide noncommercial release expected Real Soon Now
  - Release quality
    - CODE framework used day-to-day in the IPG
  - Preliminary grid management system
- Our future plans include:
  - Define and be compatible with Grid Forum standards
  - Use in the IPG (Need a web interface)
  - Develop more sensors and actuators
  - Sensors and actuators as programs as well as classes
  - More sophisticated event service
    - event routing network, more subscription models and options
  - OGSI as hosting environment
  - Work with IPG (and other) administrators to improve the grid management system
  - A public release! Open source!

GRIP: Creating Interoperability between Grids

Dietmar W. Erwin and Philipp Wieder
Central Institue for Applied Mathematice (ZAM), Research Centre Julich

Abstract

During the short history of Grid Computing several implementation of Grid solutions have been created. Most of them were started before the term Grid was coined and they often took a different approach. Globus provides an elaborate, function-rich toolkit that allows to develop Grid applications using documented interfaces and libraries. In contrast, UNICORE implemented a vertically integrated solution that allows users to create complex, interdependent multi-system and multi-site jobs giving the user seamless, secure, and intuitive access to distributed computing resources, software, and data. By design UNICORE and Globus complement each other; on the other hand there is a certain overlap in components that were developed independently by the two development teams

To exploit the respective strength of each of the systems project GRIP (Grid Interoperability Project) was proposed to the European Commission and funded under grant IST-2001-32257). The goals of project GRIP were: (1) Use the UNICORE client to access both UNICORE and Globus controlled resources in a seamless fashion. (2) Develop a cross-grid broker. (3) Demonstrate the interoperability using biomolecular and meteorological applications. (4) Contribute to working and research groups of Global Grid Forum.

The objectives of seamless job submission, control over jobs, and retrieval of results in a seamless fashion have been demonstrated at the end of the first project year. Generation of Globus proxy certificates has been integrated transparently into the client. The server component of UNICORE, the Target System Interface, has been extended to translate UNICORE constructs into Globus Resource Specification Language and to return results created by a Globus system into the UNICORE data space. Wrappers have been developed to allow to run applications, for example commercial ones, for which no source exists, not only under UNICORE but also under Globus. The experience the project gained so far will be presented

The inter-grid broker development and the collaboration in GGF is work in progress. With the advent of OGSA GRIP proposed a change in the workplan to create interoperability not only with Globus but with Grid Services in general. This change has been approved by the European Commission and a first implementation will be done in 2003.

Presentation

GRIP – Creating Interoperability between Grids

Philipp Wieder, Dietmar Erwin
Forschungszentrum Jülich
GGF8 – PGM RG
Seattle, June 26, 2003

Contents

Motivation

Moving towards THE GRID:

- Standardization
- Interoperability layer
- Applications

➡ Interoperation between UNICORE and Globus combining the unique strength of both systems

UNICORE Status

Uniform **I**nterface to **Co**mputing **Re**sources

- Project UNICORE Plus (funded by BMBF, grant: 01 IR 001) successfully completed
- UNICORE Pro software marketed and supported by Pallas
- Software available as Open Source for R&D projects: http: //www.unicore.org/download
- Basis for a German HPC Grid
- Used in several EC funded Grid projects
- Selected by Japanese NAREGI project

UNICORE Highlights

GRIP

- System-independent creation and control of jobs
- Support for multi-system and multi-site jobs
- Dynamic flow control
- Integrated security using X.509 certificates
- Access to remote file systems and archives
- Extensible support for scientific & commercial applications
- Minimal intrusion into site autonomy

Forschungszentrum Jülich
in der Helmholtz-Gemeinschaft
GGF 8 - PGM, June 2003
5

Globus Status

GRIP

- Globus Toolkit (GT) 2.4 and 3 beta
- Many GT 2.x based solutions available (IBM Grid Toolbox ...)
- Software is available as Open Source at http: //www.globus.org/gt2.4/download.html
- GT is used in projects & testbeds worldwide (DataGrid, NASA IPG ...)

Forschungszentrum Jülich
in der Helmholtz-Gemeinschaft
GGF 8 - PGM, June 2003
6

Globus Highlights

GRIP

- Set of extensible services and corresponding APIs
- Collection of commands to access services
- Services are Grid Security Infrastructure (GSI) enabled
- Commodity Grid Kits (CoG Kits) to build Grid portals
- First OGSI compliant Grid implementation

Forschungszentrum Jülich
in der Helmholtz-Gemeinschaft
GGF 8 - PGM, June 2003
7

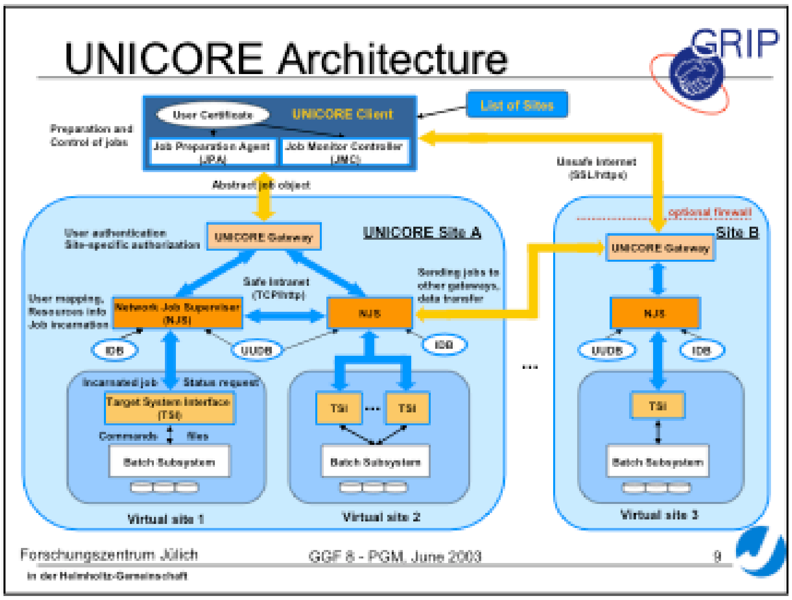
Objectives of GRIP

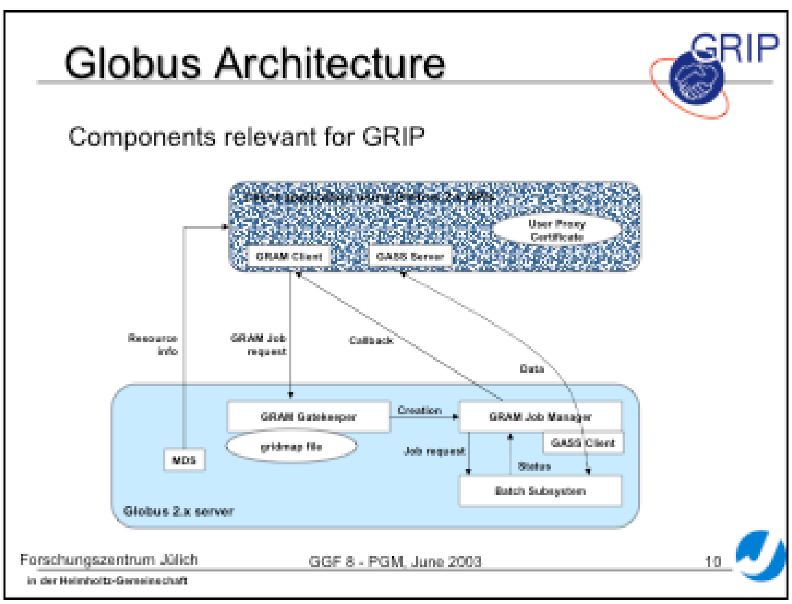
GRIP

Grid **I**nteroperability **P**roject: EU grant IST-2001-32257

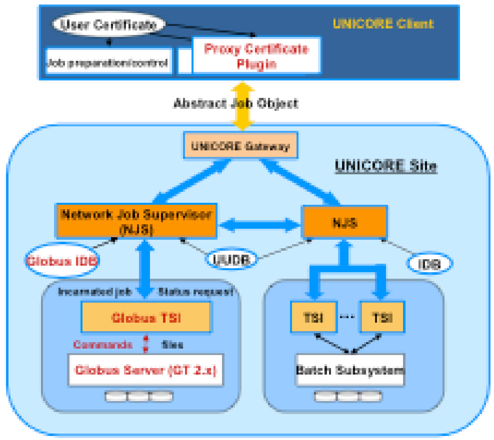
- Make Globus controlled resources available to UNICORE users
- Develop software to enable the interoperation of UNICORE and Globus
- Cross-Grid information brokerage
- Build and demonstrate biomolecular and meteorological inter-Grid applications
- Contribution to standarization efforts
- Evolve towards a service oriented Grid

Forschungszentrum Jülich
in der Helmholtz-Gemeinschaft
GGF 8 - PGM, June 2003
8


UNICORE Architecture
GRIP
User Certificate
UNICORE Client
List of Sites
Preparation and Control of jobs
Job Preparation Agent (JPA)
Job Monitor Controller (JMC)
Unsafe Internet (SSL/https)
Abstract job object
optional firewall
User authentication
Site-specific authorization
UNICORE Gateway
UNICORE Site A
Site B
UNICORE Gateway
Safe intranet (TCP/http)
Sending jobs to other gateways, data transfer
User mapping, Resources info Job incarnation
Network Job Supervisor (NJS)
NJS
NJS
IDB
UUDB
IDB
UUDB
IDB
...
Incarnated job
Status request
Target System Interface (TSI)
TSI
TSI
TSI
Commands
files
Batch Subsystem
Batch Subsystem
Batch Subsystem
Virtual site 1
Virtual site 2
Virtual site 3
Forschungszentrum Jülich
in der Helmholtz-Gemeinschaft
GGF 8 - PGM, June 2003
9


Globus Architecture
GRIP
Components relevant for GRIP
User Proxy Certificate
GRAM Client
GASS Server
Resource info
GRAM Job request
Callback
Data
GRAM Gatekeeper
Creation
GRAM Job Manager
GASS Client
gridmap file
Job request
Status
MDS
Batch Subsystem
Globus 2.x server
Forschungszentrum Jülich
in der Helmholtz-Gemeinschaft
GGF 8 - PGM, June 2003
10

GRIP Architecture



Interoperability Components

Discussed in this presentation:

- Security
- Resource provisioning & brokerage
- Applications support

Others:

- Resource description mapping
- Job submission & monitoring
- Data transfer

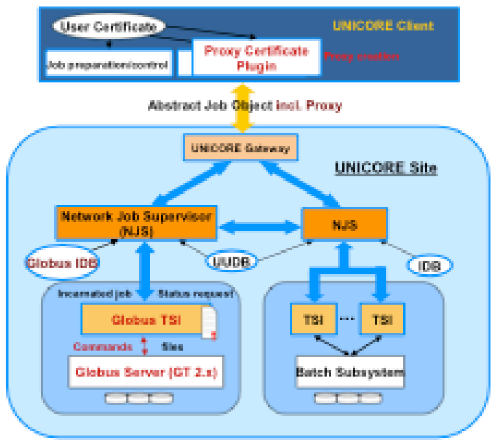
Security

- Both systems use PKIs utilizing X.509 certificates
- UNICORE
  - End-to-end security
  - Certificates for user & server authentication and signing of jobs
  - Authorization entity: UNICORE user database
- Globus
  - Proxy delegation
  - Proxies are used for user & server authentication
  - Authorization entity: gridmap-file

Security (cont)

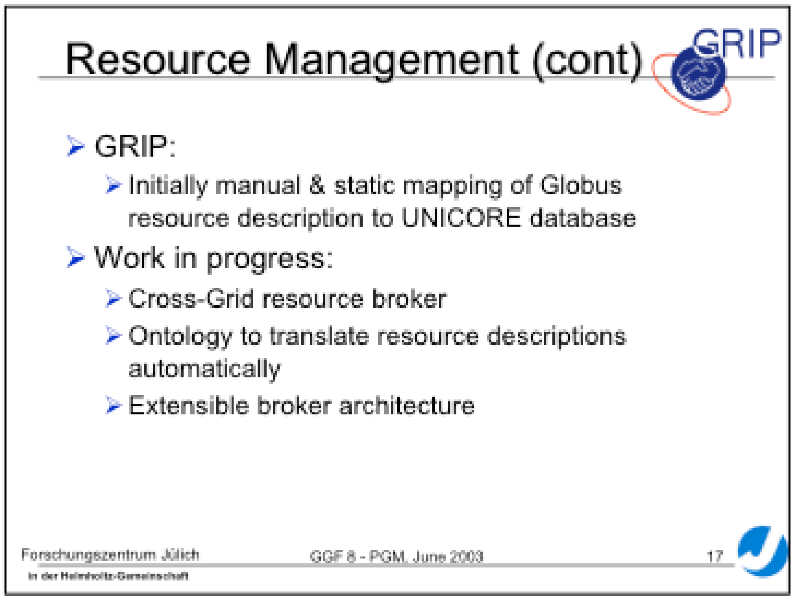
- GRIP
  - Client generates proxy certificates automatically
  - Proxy transferred via SSL within job (signed), not stored in between
  - Proxy stored in user's filespace
  - Proxy is removed after job execution
  - Proxies are signed by GRIP CA
  - Globus site must trust GRIP CA

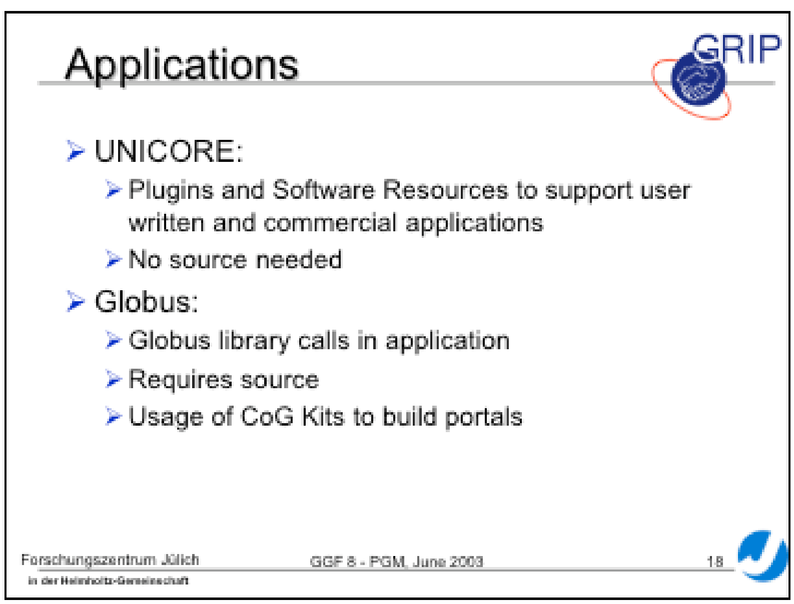
Security (cont)



Resource Management

- UNICORE
  - Distributed database (IDB) for each target system
  - Resource info automatically provided to client
  - Software is also a resource in the database
- Globus
  - Monitoring & Discovery Service (MDS) using LDAP
  - Client API to query MDS
  - No software resources

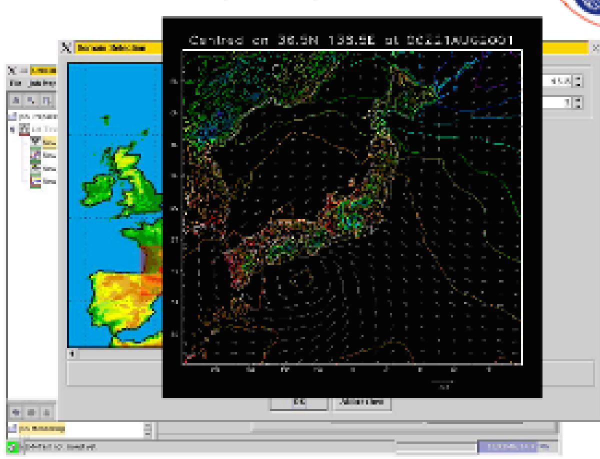

Resource Management (cont)
GRIP
GRIP:
Initially manual & static mapping of Globus resource description to UNICORE database
Work in progress:
Cross-Grid resource broker
Ontology to translate resource descriptions automatically
Extensible broker architecture
Forschungszentrum Jülich
in der Helmholtz-Gemeinschaft
GGF 8 - PGM, June 2003
17


Applications
GRIP
UNICORE:
Plugins and Software Resources to support user written and commercial applications
No source needed
Globus:
Globus library calls in application
Requires source
Usage of CoG Kits to build portals
Forschungszentrum Jülich
in der Helmholtz-Gemeinschaft
GGF 8 - PGM, June 2003
18

Applications (cont)

- GRIP:
  - Wrappers for applications to allow them to execute transparently either on Globus or on UNICORE
  - No source needed
  - Hopefully only a temporary solution till applications become Grid Services

Applications (cont)



GRIP Issues & Challenges

- Administration
- Management
- Support
- Interoperability Challenges

Administration

- No major additional administration for UNICORE & Globus sites
- Maintain UNICORE user database to authorize access to Globus Virtual site
- Globus site installs a Globus TSI on a dedicated system or the target system
- Enable GSI to use GRIP CA signed proxies

Management

- Management is distributed in UNICORE and GRIP
- Organizations retain their autonomy
- Coordination is a must
  - Acceptance of CAs
  - Expiration of certificates (esp. for servers)
  - Management of user mappings
  - Interoperability between software versions
  - Dependency on other software (e.g. Java version)
- Additional: Dependency on Globus versions and policies

Support

- Professional support and enhancements are essential for production
- Pallas is providing this for UNICORE for administrators and second level support for users
- Open Source is necessary but not enough
- User support through HPC centers

GridWeaver – Large-Scale Fabric Management for Grid Computing

Peter Toft, Guillaume Mecheneau and Patrick Goldsack, HP Labs, George Beckett and Kostas Kavoussanakis, EPCC, Paul Anderson and Jim Paterson, University of Edinburgh

Abstract

1 The GridWeaver Project

The GridWeaver project is a collaboration between Edinburgh University School of Informatics, HP Labs in Bristol, and EPCC, as part of the UK e-Science Core Programme. GridWeaver is focused on the research challenges of managing the configuration of very large-scale computational infrastructures, especially those that form parts of computing Grids. Our goal is to develop approaches and technologies that are capable of representing and realising complex system configurations of hardware, software and services, across very large-scale infrastructures incorporating high degrees of heterogeneity, dynamism and devolved control. More information can be found at the project website www.gridweaver.org.

The talk for the PGM-RG Workshop will cover the following context, research problems, and concrete areas of progress in the project.

2 Preamble: A Paradigm Shift for Computing

The drive towards Grid-based computing is one leading indicator that computing infrastructure is in the initial phases of its next paradigmshift – where a new, over-arching computing model replaces the previous one. As computer hardware has become more powerful, cheaper and smaller, it is proliferating to an extraordinary degree. And there realising the new paradigm raises many interesting research problems. Grid computing research and development is attacking many of the the upper layers of the problem space, providing ways to discover and utilise appropriate resources in a global-scale computing infrastructure. The focus of the GridWeaver Project is on the lower levels of the problem space, dealing with complex system configuration. The ultimate goal of our research is to provide fabric configuration and management solutions that are capable of handling the full complexity and diversity found in emerging computing fabrics, and which are able to automate many node-centric activities that would today need to be performed by people. Such solutions will be capable of (for example) automatically configuring a complex Grid fabric from the ground up, to permit its participation as an element in a global Grid.

3 Research Challenges

We see three key categories of research challenge:

1. Representing system configurations;
2. Using and manipulating system configurations;
3. Deploying system configurations.

3.1 Representing system configurations

We believe that in order to manage complex system configurations, it is first desirable to have a language-based system that allows us to express, manipulate, and reason about, all relevant aspects of the the configuration. We further believe that this goal is best served by adopting a declarative language for expressing configuration. Given this, research challenges include:

- The basic design of the language itself: we require a language that is able to express all the necessary relationships between the configuration parameters that form part of our system configurations. The language does not itself embody the models we are representing, but it must be capable of representing all of the required attributes of the models.

- Representing a wide spectrum of abstraction levels and levels of detail: we must be able to express levels of detail and abstraction varying from very low-level device configuration parameters up to large-grained entities such as clusters. This requires that we be able to compose system configurations by building up highabstraction components from lower level ones.

- Representing a rich range of dependencies between configuration parameters: this is vital in order, for example, to support relationships between clients and

    - Broad device and application support: in order to configure devices or software components, the runtime deployment system must know how to configure these elements. Typically, this requires that components be written that are capable of interpreting configuration parameters for a device or application, and making the appropriate configuration changes.

    - Proxying: some devices will not have the required capabilities to be managed by the configuration system, and will need to managed by proxies that can make the appropriate control translations

4 GridWeaver Research

GridWeaver will not solve, nor even work on, all of the research challenges listed above. We are currently focusing on a subset of interesting and challenging problems, including: • Development of a suitable language system (syntax and language tools) for representing system configurations; • Development of an experimental architecture for the runtime configuration deployment system; • Design and development of a small number of system components to explore the use of the language and to test the architecture; • Demonstration of the use of this approach by configuring a fabric to support a challenging application. This is currently focused on automatically configuring a Globus-enabled fabric from bare compute nodes. We have made significant progress on these strands of research, by bringing together two technologies: “SmartFrog” from HP Labs, and “LCFG” from the University of Edinburgh. We will discuss how these technologies are being used to meet the research goals above.

Presentation

GridWeaver

Large-Scale, Adaptive Fabric Configuration for Grid Computing

Peter Toft
HP Labs, Bristol

June 2003 (v1.02)
Localised for UK English

The GridWeaver Project

- A collaboration between
  - Serrano Project, HP Labs Bristol (UK)
  - School of Informatics, University of Edinburgh
  - EPCC, University of Edinburgh
- Combining research interests and technologies from HP Labs (SmartFrog) and Edinburgh (LCFG)
- Funded by the UK e-Science Grid Core Programme, and by HP
  - A 1 year project, to July '03
  - Research-oriented, not building production systems

GridWeaver 2

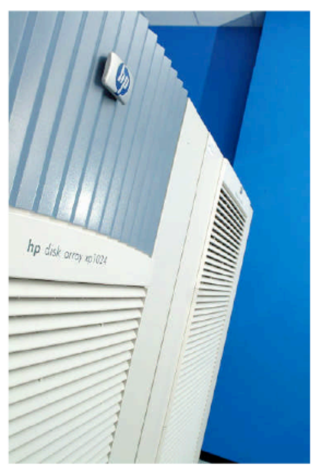
The Challenge

- An effective Grid assumes the existence of correctly operating, large-scale fabrics
- Every aspect of the fabric must be correctly configured
  - From base OS on a single fabric element, up to complex, coupled, distributed services
- Challenging problems:
  - Scale
  - Diversity
  - Complexity
  - Dynamism
- Aim to:
  - Describe whole-fabric configuration
  - Deliver a correctly-configured fabric
  - With automatic adaptive behaviour

GridWeaver 3

Research Interests

- Expressing system configurations
  - Models and languages for representing configurations of resources
- Using and manipulating system configurations
  - Tools to assist in creating and manipulating correct configurations
- Deploying system configurations
  - Turning the representation of your desired system into a realised, running system
  - Keeping the configuration correct over time
- Creating adaptive system behaviour
  - A framework for automatic reconfiguration to accommodate changes (including failures)



GridWeaver 4

Technologies: LCFG and SmartFrog

- A common philosophy …
  - Language-based approaches for expressing whole-system configuration
  - Frameworks and extensible component sets for realising system configurations
  - "Asymptotic" configuration to deal with scale
- … with complementary foci
  - LCFG focuses on configuring and managing individual nodes in a fabric
  - SmartFrog focuses on configuring and orchestrating distributed applications running across nodes

GridWeaver 5

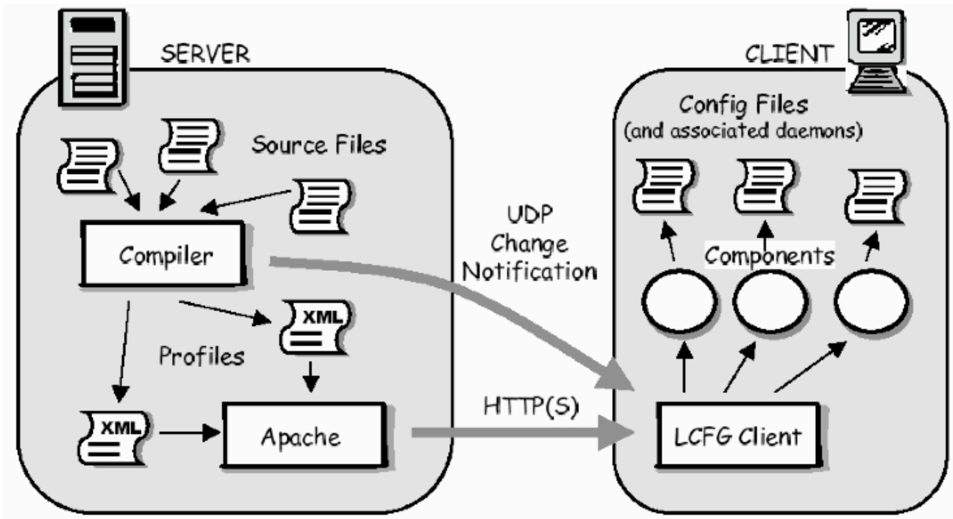
Overview of LCFG

LCFG: "Local Configuration System"

- Declarative, prototype-based language for defining fabric configurations
  - Configuration "aspects" are combined to create a complete configuration profile for each node
- Components for configuring and managing most aspects of node configuration
  - E.g.: configuring, starting, and stopping local services
- Runtime framework for deploying configurations via components
- Centrally-controlled fabric management: configuration server defines the required fabric configuration
- Bare metal installation
- Currently Linux-oriented

GridWeaver 6

Overview of LCFG

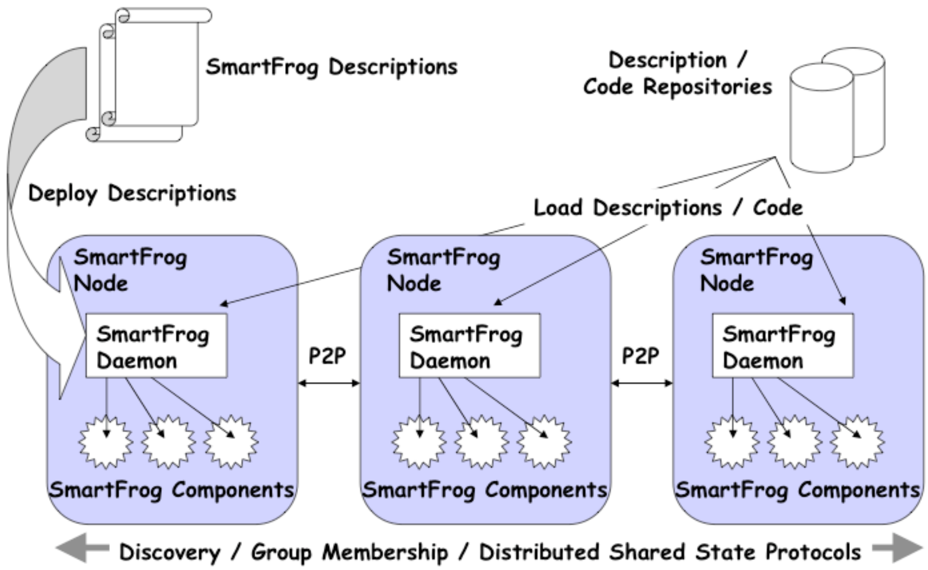


Overview of SmartFrog

SmartFrog: "Smart Framework for Object Groups"

- Describes, deploys and orchestrates distributed, component-based applications
- Language
  - Declarative, prototype-based, attribute description language, supporting templates, composition, late attribute binding, predicates (soon), etc.
- Distributed, runtime deployment infrastructure
  - Secure (certificate-based) deployment of descriptions and code
  - Multiple methods of loading descriptions and code
- Component model and configurator components
  - A defined component lifecycle for each component
  - Extensible set of components developed for grouped lifecycles, workflows, flexible binding mechanisms, etc.
- No central point of control required; peer-to-peer interactions
- Portable across many classes of fabric element (written in Java)

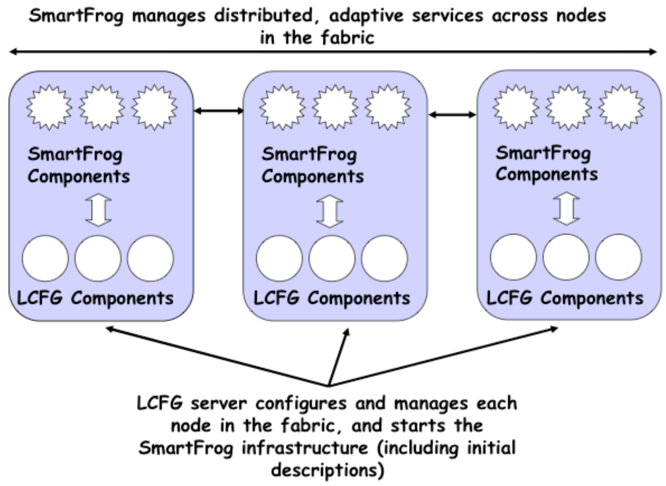
Overview of SmartFrog



Combining LCFG and SmartFrog

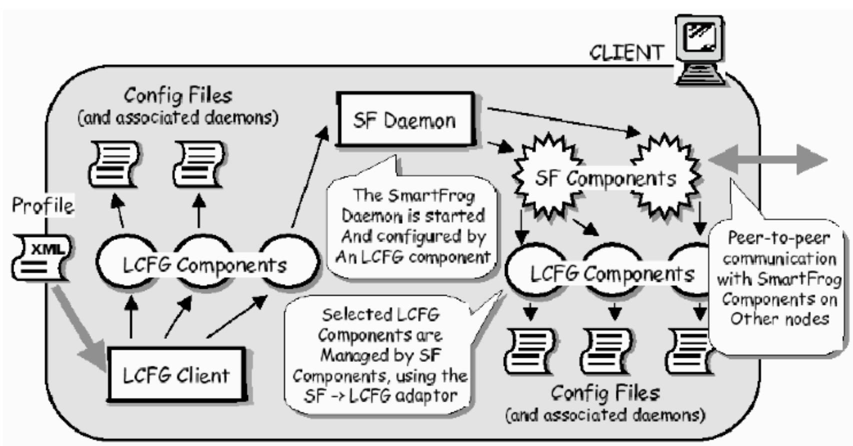
- Division of labour
  - LCFG: per-node configuration, bare-metal upwards
  - SmartFrog: higher-level, distributed, adaptive services
- Integrated configuration infrastructure
  - LCFG configures, starts and manages the SmartFrog daemon on each node
  - SmartFrog controls LCFG components using a generic SmartFrog / LCFG adaptor
- Planned (but not yet done): unified description language (using SmartFrog language v2)
  - Complete fabric description using one, powerful representation
  - “Compilation” results in LCFG node configuration profiles plus deployable SmartFrog descriptions

Combining LCFG and SmartFrog



GridWeaver 11

Combining LCFG and SmartFrog



GridWeaver 12

GridWeaver Prototype

- Bare-metal OS installation, configuration and ignition
  - ? Basic fabric management
- Automatic installation and configuration of a Globus GT3 (OGSI) infrastructure
  - ? Grid-enabled fabric
- GPrint: an example adaptive, cross-fabric print service
  - ? Fabric + adaptive service
- Exposure of GPrint as a Grid Service via Globus
  - ? Grid-enabled service

GridWeaver 13

Video Sequence

GridWeaver 15

Future Work?

- From a research prototype to a production system
  - LCFG is open source (GPL)
  - SmartFrog release to open source (LGPL) in '03
  - Work needed to create a portable, production-quality system
- Grid Configuration Service Interface
  - A Grid Service interface for fine-grained control over aspects of fabric configuration
- Many other interesting research problems
  - Representing time-based properties in configurations
  - Devolving control of different configuration aspects, securely
  - …

GridWeaver 16

Project Reports

Technologies for Large-Scale Configuration Management

Abstract

Acknowledgements

GridWeaver

Report 1: "Technologies for Large-Scale Configuration Management"

December 2002

Report 2: "Experiences and Challenges of Large-Scale System Configuration"

March 2003

Experiences and Challenges of Large-Scale System Configuration

Abstract

GridWeaver

GridWeaver 17

More Information

- Contacts
  - HP: Peter Toft (peter.toft@hp.com)
  - University of Edinburgh: Paul Anderson (dcspaul@inf.ed.ac.uk)

- Web pages
  - GridWeaver: www.gridweaver.org
  - SmartFrog: www-uk.hpl.hp.com/smartfrog www.smartfrog.org
  - LCFG: www.lcfg.org

- Paper in LISA '03: "SmartFrog meets LCFG: Autonomous Reconfiguration with Central Policy Control"

- Further project reports available in August '03 (covering modelling and language, integration architecture, prototype design)

GridWeaver 18

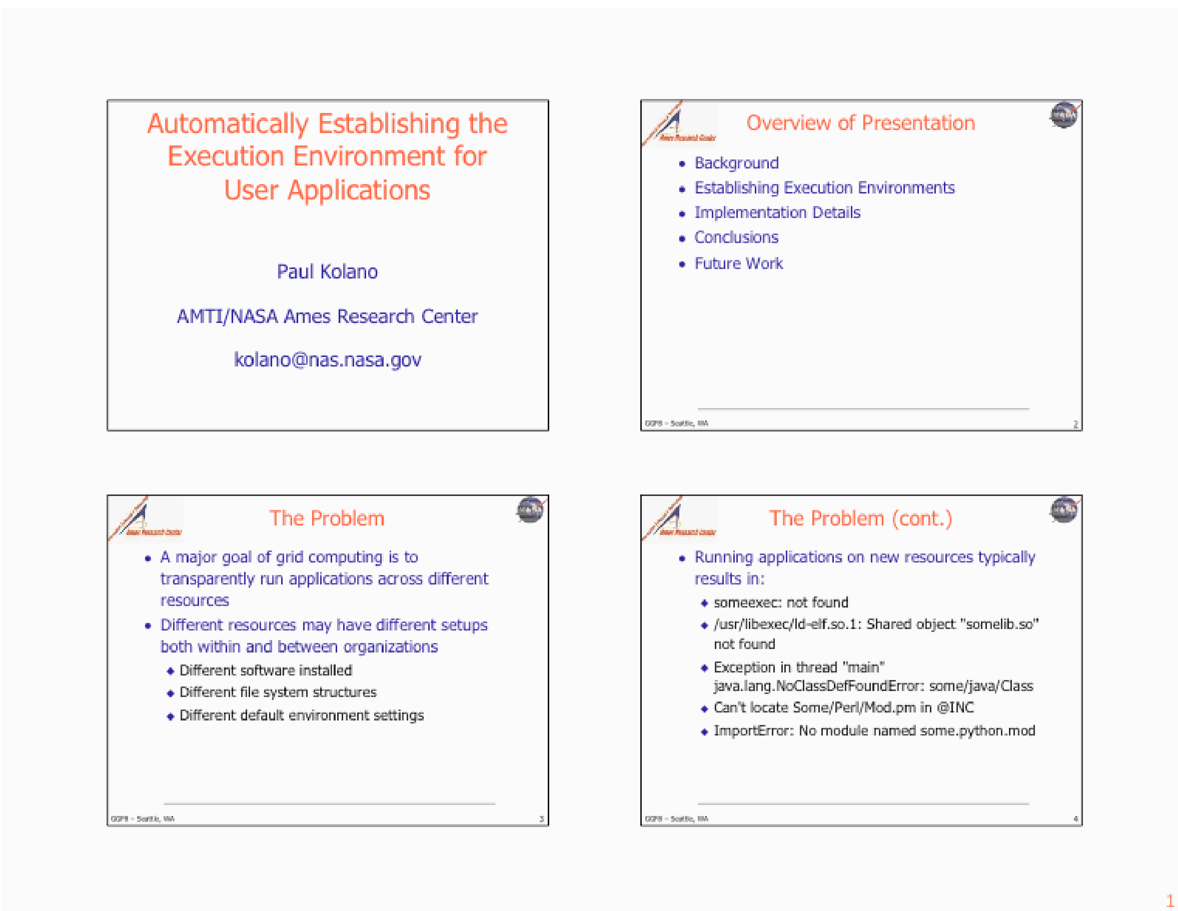
Automatically Establishing the Execution Environment for User Applications

Paul Kolano, AMTI/NASA Ames Research Center

Abstract

Grid computing promises the ability to connect geographically and organizationally distributed resources to increase effective computational power, resource utilization, and resource accessibility. For grid computing to be successful, users must be able to easily execute the same application on different resources. Different resources, however, may be administered by different organizations with different software installed, different file system structures, and different default environment settings. Even within the same organization, the set of software installed on a given resource is in constant flux with additions, upgrades, and removals. Users cannot be expected to understand all of the idiosyncrasies of each resource they may wish to execute jobs on, thus must be provided with automated assistance. In this talk, we describe a new OGSI-compliant grid service that has been implemented as part of NASA's Information Power Grid (IPG) project to automatically establish the execution environment for user applications.

Presentation



The Problem (cont.)

- Users end up wasting time setting up the resources that were supposed to save them time

GGF8 – Seattle, WA 5

The Goal

- Automatically ensure that user applications will not encounter software dependency failures during execution

GGF8 – Seattle, WA 6

Typical Solutions

- Statically-linked executables
  - Result in
    - Overly large executables
    - Inefficient use of memory
    - Hard-coding library bugs into code
- Custom software packages
  - Require detailed knowledge of
    - Dependency analysis techniques
    - Differences in environment settings
      - Operating systems
      - Software types

GGF8 – Seattle, WA 7

Typical Solutions (cont.)

- Waste time and allocations better spent on actual work
  - Transferring unnecessarily large files
  - Manually preparing custom packages

GGF8 – Seattle, WA 8

2

Related Projects

- Globus Executable Management (GEM)
  - Copy appropriate executable from network repository
- UNICORE
  - Transform abstract executable names to absolute paths
- Automatic Configuration Service (F. Kon et al.)
  - Automatically install software as necessary for component-based applications using manually specified dependencies
- Installers, Package Managers, and Application Management Systems
  - Provide consistent set of software on one or more systems
- Replica Management Systems
  - Facilitate location, selection, and replication of datasets

GGF8 – Seattle, WA 9

Related Projects (cont.)

- None have automatic dependency analysis
- Most have no treatment of environment variables
- Most only support executables
- Some require significant administration
- Some can't dynamically install new software

GGF8 – Seattle, WA 10

Summary

- Existing approaches do not provide enough assistance
- This work describes a new grid service for automatically establishing execution environments

GGF8 – Seattle, WA 11

Establishing Execution Environments

- Determine software the application requires
- Provide location for software on execution host
  - Determine if software is already installed
  - Find a source for missing software
  - Copy missing software to execution host
- Set environment variables based on locations

GGF8 – Seattle, WA 12

3

Establishing Execution Environments

- Don't want service to transfer software itself
  - User may cancel job
  - Previous job operations may fail
- Instead, add file operations and modify execution operation environment settings

GGF8 – Seattle, WA 13

Jobs

- Set of operations arranged in some topology
- File Operations (minimum requirements)
  - Copy files
  - Create directories
- Execution Operations (minimum requirements)
  - Host to execute on
  - Path of application to execute
  - Environment mapping from variables to values

GGF8 – Seattle, WA 14

Example Job

```
// Example.java

import org.apache.commons.logging.
            impl.SimpleLog;
import my.TimeClass;

public class Example {
    public static void main(String argv[]) {
        SimpleLog log =
            new SimpleLog("output");
        TimeClass tc = new TimeClass();
        if (tc.isTimeEven()) {
            log.info("even");
        } else {
            log.info("odd");
        }
    }
}
```

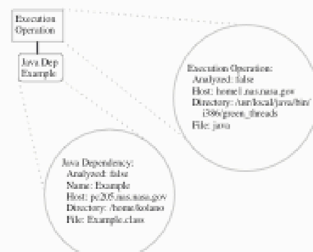
```
// TimeClass.java

package my;

public class TimeClass {
    public boolean isTimeEven() {
        long time =
            System.currentTimeMillis();
        if (time % 2 == 0) return true;
        else return false;
    }
}
```

GGF8 – Seattle, WA 15

Example Job



GGF8 – Seattle, WA 16

4

Stage 1: Dependency Analysis

- Determine application software dependencies
- Support common software types
- Concentrate initially on statically generated dependencies (i.e. do not worry about cases such as char *lib = f(); dlopen(lib))

Dependencies

- Basic information
  - Type
    - e.g. Executable
  - Name
    - e.g. w3m
  - Version range
    - e.g. [3.0, 3.1.1]
  - Feature list
    - e.g. compiled with SSL support
- Extended information
  - Source host
  - Source path
  - Target path
  - Analyzed flag

Dependencies (cont.)

- Currently supported types
  - Executable and Linking Format (ELF) objects
    - Executables
    - Shared libraries
  - Java classes
  - Perl programs
  - Python programs

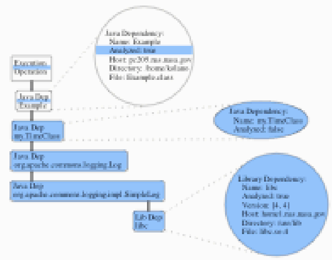
Stage 1: Dependency Analysis (cont.)

- Dependency information is embedded
  - ELF executables and libraries
  - Java classes
- Dependency information must be derived
  - Perl programs
  - Python programs
  - Techniques
    - Textually traverse for relevant expressions
    - Partially evaluate using interpreter mechanisms

Stage 1: Dependency Analysis (cont.)

- Use as many existing tools as possible
  - Executables and libraries
    - ldd, elfdump
  - Java classes
    - com.sun.jini.tool.ClassDep
  - Perl programs
    - Module::ScanDeps
  - Python programs
    - modulefinder

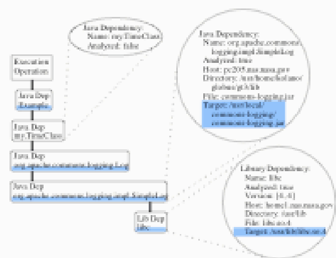
Example Job After Dependency Analysis



Stage 2(a): Dependency Location

- Determine existing location of software on target system
- Minimize number of files to transfer
- Searching an entire file system is impractical
- Must limit search space to specific paths
  - Which paths?
    - Add paths based on Filesystem Hierarchy Standard
    - Add paths based on user and system-default settings
  - Cannot guarantee file will be found in all cases
- Find using ls and Java, Perl, and Python interpreters

Example Job After Dependency Location

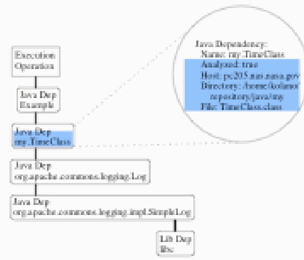


Stage 2(b): Dependency Lookup

- Find a source for missing software
- Find dependencies of missing software
- Use software catalog
  - Contains mappings from LFNs to PFNs
  - LFNs based on dependency name, type, supported operation system, and version
  - Contains dependencies of each PFN
  - Allows both centrally-managed and user-defined mappings

GCF8 – Seattle, WA 25

Example Job After Dependency Lookup



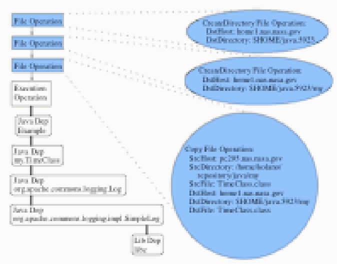
GCF8 – Seattle, WA 26

Stage 2(c): Dependency Transfer

- Copy missing software to execution host
- Must create correct directory hierarchy for Java, Perl, and Python software
  - e.g. my.TimeClass.class can only be found if it exists in some directory as .../my/TimeClass.class
- Copy at most once per job

GCF8 – Seattle, WA 27

Example Job After Dependency Transfer



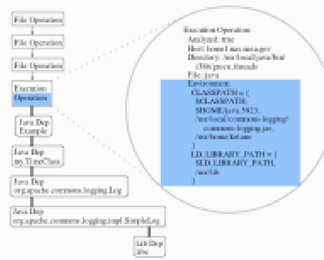
GCF8 – Seattle, WA 28

Stage 3: Variable Setup

- Application must be able to locate software
- Set environment variables to find existing and soon to be existing software
- Must consider directory hierarchy for Java, Perl, Python software
  - e.g. for my.TimeClass at /somedir/my/TimeClass.class, CLASSPATH must contain /somedir

GGF8 – Seattle, WA 29

Example Job After Variable Setup



GGF8 – Seattle, WA 30

Possible Failures

- Application depends on A, but A cannot be located anywhere
- Application depends on A, which depends on B, but analysis techniques used on A are inadequate to determine B is a dependency
- Application does not depend on A, but analysis techniques report A is a dependency

GGF8 – Seattle, WA 31

Dealing with Failures

- Notify user to prevent wasted effort
- Missing software
  - Provide convenience methods to locate relevant dependencies after transformation
- False negatives and false positives
  - Cannot detect automatically
  - Currently, only Perl analysis is susceptible
  - Provide user with flexibility to compensate when necessary

GGF8 – Seattle, WA 32

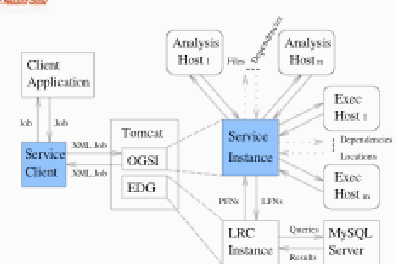
Flexibility

- User has complete control of job transformation
  - Can execute stages individually
  - Can specify dependencies manually
  - Can turn analysis off for individual items
  - Can specify an exact source for software
  - Can specify an existing location on execution host
  - Can manage personal software catalog

Implementation

- Implemented in Java and Bourne shell scripts
- Runs as an OGSI-compliant grid service
- Uses OGSI GRAM service to execute analysis and location scripts
- Uses European DataGrid Local Replica Catalog as software catalog

Implementation (cont.)



Where does this fit in?

- NASA Information Power Grid (IPG)
  - Current prototype services
    - Resource Broker
      - Select resources for jobs based on user constraints
    - Job Manager
      - Reliably execute jobs on specific resources
  - Establish environment after selection and before execution

Implementation (cont.)

- Service exists in prototype form
- All discussed functionality fully tested on FreeBSD
- Analysis and location scripts fully tested on FreeBSD, IRIX, and SunOS
- Waiting for full IPG deployment
  - GT3 stability and IRIX support
  - RB and JM are GT2 services and not yet OGSI-compliant

Conclusions

- Implemented a new OGSI-compliant service with functionality for
  - Automatically identifying application dependencies
  - Managing a flexible software catalog used as a source for key software
  - Establishing a suitable environment by transferring dependent software and setting environment variables
- Increases pool of compatible resources with little or no user intervention
- Net result is increase in user productivity

Future Work

- Software caching
- Additional dependency types
- Additional analysis capabilities
- Full IPG deployment
- Advanced software installation mechanisms
- Full version and feature support

Exegrid - Workflow Support and Resource Management

Jurgen Falkner, Fraunhofer Resource Grid, Fraunhofer IAO

Abstract

Based on a newly developed Grid Application Definition Language Fraunhofer has developed software to manage complex workflows on grids. Based on tge very general notion of petri nets*, exegrid allows an easy development of applications on tge grid and supports automatic resource brokerage. Exegrid will become free software within the next two months. www.fhrg.fhg.de

Presentations

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Profile of the Fraunhofer-Gesellschaft

Founded in: 1949

Staff of 12 000 (2002)

Research volume of 1 billion Euro (2002)

57 research establishments, operating as independent »Profit-Centers«

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Locations in Germany

Aachen, Berlin, Braunschweig, Bremen, Chemnitz, Cottbus, Darmstadt, Dortmund, Dresden, Duisburg, Efringen-Kirchen, Erlangen, Euskirchen, Frankfurt, Freiburg, Freising, Golm, Halle, Hannover, Holzkirchen, Ilmenau, Itzehoe, Jena, Kaiserslautern, Karlsruhe, Magdeburg, München, Nürnberg, Oberhausen, Paderborn, Pfinztal, Rostock, Saarbrücken, Sankt Augustin, Sankt Ingbert, Schmallenberg, Stuttgart, Teltow, Wertheim, Würzburg

Fraunhofer Gesellschaft

57 Institutes

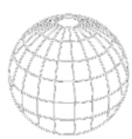
mainly engineering

Production technology
IT
Microelectronics
Material Sciences
Energy and Building
Life Sciences

12000 Scientists

Fraunhofer Institut Arbeitswirtschaft und Organisation

International locations



Fraunhofer USA
Boston (Massachusetts), Pittsburgh (Pennsylvania),
Plymouth (Michigan), Providence (Rhode Island),
Newark (Delaware), College Park (Maryland), Peoria (Illinois)

Fraunhofer Asia
Beijing (China), Singapore, Jakarta (Indonesia), Tokyo (Japan)

Fraunhofer Institut Arbeitswirtschaft und Organisation

Research fields

- Materials technology, component behaviour
- Production technology, manufacturing engineering
- Information and communications technology
- Microelectronics, microsystems technology
- Sensor systems, testing and measurement technology
- Process technology
- Energy and building technology
- Environmental and health research
- Technical and economic studies
- Information transfer

Director

Univ. Prof. Dr.-Ing. Dieter Spath

Profile

Founded in: 1981

Staff of 140 (2002)

21,5 million Euro research volume (2002)

Location: Stuttgart

Website: www.iao.fraunhofer.de

Business areas of the institute

- Innovative Work Environments and Human Resources Customer Communication and Care
- Enterprise Information Systems
- E-Business
- Multimedia Systems and Learning Environments
- Enterprise Strategies and Business Processes
- Virtual Engineering
- Integrated Software Development
- Development and Management of Services
- **Grid Computing**

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Demo Centers (selection)

- Electronic Business Innovation Center
- Engineering Solution Center (ESC)
- Immersive Driving Simulator
- Knowledge Media Lab
- E-Factory
- New Media Communication Lab (NMC)
- Office Innovation Center (OIC)
- 6-sided-CAVE »HyPI 6«

Page 10

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Cooperation

Fraunhofer IAO and the Institute for Human Factors and Technology Management (IAT) of the University of Stuttgart work in close cooperation

- Basic research at the university, application-oriented science and economic practice are combined
- Practical implementation in industrial projects
- Close cooperation due to proximity and shared projects

Page 11

eXegrid

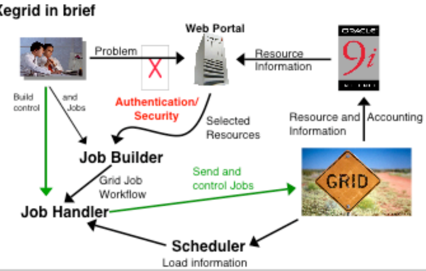
An open source approach for the industrial use of grids

Main assumptions

- Grid Job can be a very complex workflow
  (there is a difference between grid jobs and common jobs)
- No user interaction for job scheduling and data transfer
- Mapping between software and hardware must be automated
- Need for a Grid Application DescriptionLanguage → GADL
- Job submission as easy as possible

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

eXegrid in brief

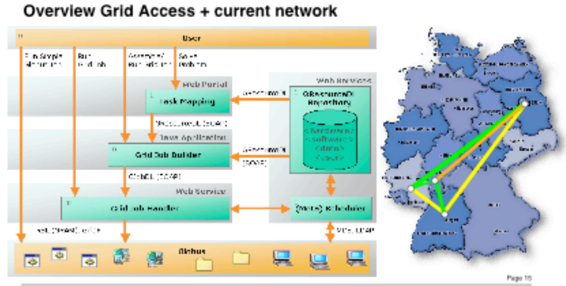


Page 14

Fraunhofer Institut Arbeitswirtschaft und Organisation

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Overview Grid Access + current network



Page 15

Fraunhofer Institut Arbeitswirtschaft und Organisation

Grid Application Description Language (GADL)

GADL	Set of XML-based description languages needed to define and to execute grid applications
	The GADL consists of:
GResourceDL	Description of resources
GJobDL	Description of grid jobs → Set of resources + workflow
GInterfaceDL	Interface definition of software components
GDataDL	Description of data

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

GADL: Grid Resource Description Language (GResourceDL)

GResourceDL	Description language for categorization and description of resources
	Used to select suitable resources to solve a given problem (task mapping)
	Definition of dependencies between resources
	Extension of resource descriptions with inheritance allows formulation of recursive descriptions
Everything is a resource!	Software components
	Hardware resources
	Measuring devices
	Data

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

GResourceDL example that depends on other resources

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE fhrgResources (View Source for full doctype...)>
<fhrgResources>
 <resource id="sleep" type="software">
 ...
 <dependencies type="depends">
  <resourceRef id="linux" type="softwareClass" />
  <resourceRef id="glibc6" type="softwareClass" />
  <resourceRef id="x86" type="hardwareClass" />
 </dependencies>
 ...
</fhrgResources>
```

depends, conflicts, provides, suggests

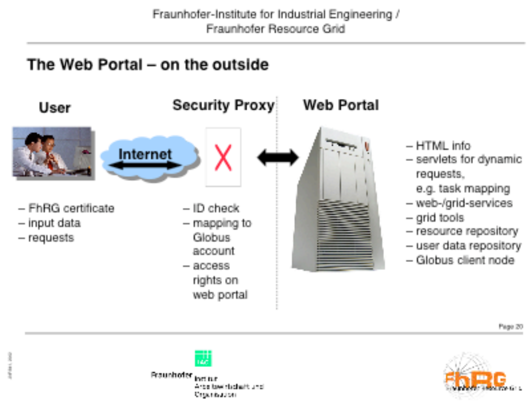
Page 18

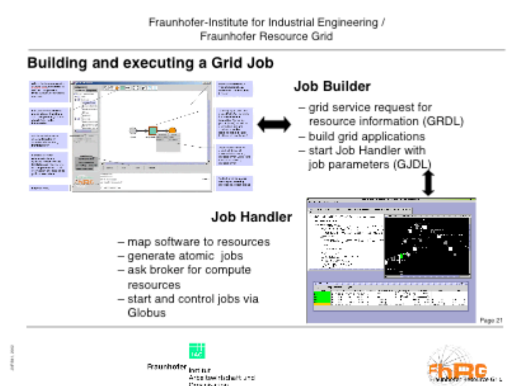
Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

GResourceDL example that provides other resources

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE fhrgResources (View Source for full doctype...)>
<fhrgResources>
 ...
 <resource id="de-fhrg-first_harlekin" type="hardware">
  <dependencies type="provides">
   <resourceRef id="x86" type="hardwareClass" />
   <resourceRef id="network-ethernet-100" type="hardwareClass" />
   <resourceRef id="linux-kernel-2-4-18" type="softwareClass" />
   <resourceRef id="glibc6" type="softwareClass" />
  </dependencies>
  <authorization>
   <userGroup id="all" read="true" write="false" />
   <userGroup id="first" read="true" write="true" execute="true" />
  </authorization>
 </resource>
</fhrgResources>
```

Page 19


Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid
The Web Portal – on the outside
User
Security Proxy
Web Portal
Internet
– FhRG certificate
– input data
– requests
– ID check
– mapping to Globus account
– access rights on web portal
– HTML info
– servlets for dynamic requests, e.g. task mapping
– web-/grid-services
– grid tools
– resource repository
– user data repository
– Globus client node
Page 20
Fraunhofer
FhRG


Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid
Building and executing a Grid Job
Job Builder
– grid service request for resource information (GRDL)
– build grid applications
– start Job Handler with job parameters (GJDL)
Job Handler
– map software to resources
– generate atomic jobs
– ask broker for compute resources
– start and control jobs via Globus
Page 21
Fraunhofer
FhRG

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Petri Nets

○	Places	Files, buffers, control places (active, done, ...)
□	Transitions	Software components, control transitions
○→□	Arcs from places to transitions (Place is input place of transition)	
□→○	Arcs from transitions to places (Place is output place of transition)	
●	Tokens	Data
Rules		A transition is activated if all input places are filled with tokens and all output places have not reached their maximum capacity of tokens
Refinement		A transition can be replaced by a Petri Net

Page 22

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Control and Data Flow as a Petri Net

○	Control place
□	Control transition
⬡	Data place (linked to a file)

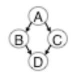
Software transition (linked to a software component)

Page 23

GADL: Grid Job Definition Language (GJobDL)

Problem — Description of complex workflows of grid jobs

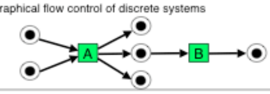
DAG



Directed Acyclic Graph (see e.g. Condor)
no bidirectional coupling (interaction)
no loops

```
PARENT A CHILD B C
PARENT B C CHILD D
```

Petri Nets — Graphical flow control of discrete systems

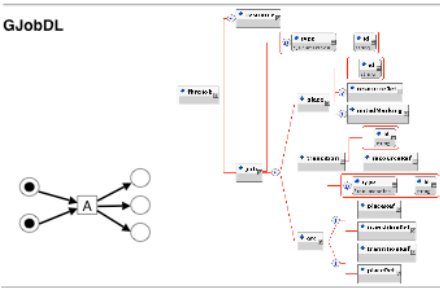


Description of Petri Nets in XML

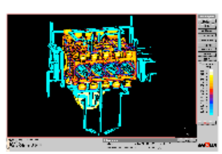
PNML — Petri Net Markup Language
(modified from *Jüngel, Kindler, Weber; HU-Berlin*)

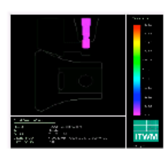
```
<place id="start">
  <initialMarking>
    <value type="boolean" op="eq">true</value>
  </initialMarking>
</place>
<place id="output"/>
<transition id="program"/>
<arc type="P2T" id="arc1">
  <placeRef id="start" />
  <transitionRef id="program" />
</arc>
<arc type="T2P" id="arc2">
  <transitionRef id="program"/>
  <placeRef id="output" />
</arc>
```

GJobDL



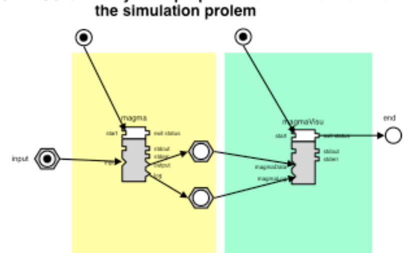
MAGMAsoft **Simulation Environment for foundries**
a commercial application



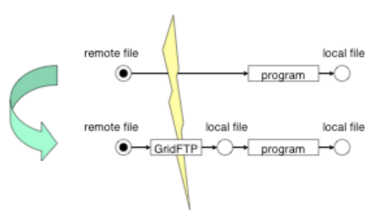


A simple example of FhRG usage

MAGMAsoft : Use your laptop and some remote site to solve the simulation prolem

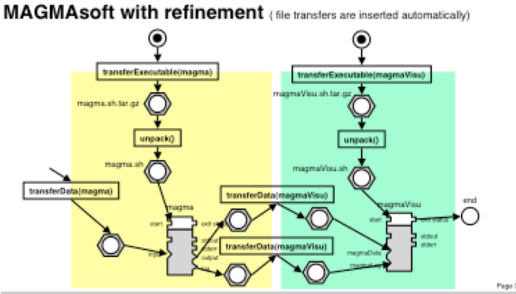


GridFTP with Petri Nets using refinement



Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

MAGMAsoft with refinement (file transfers are inserted automatically)



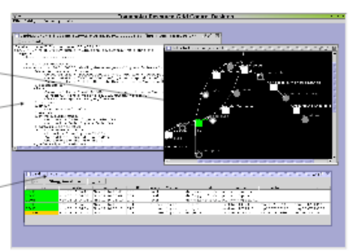
Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Tools that help

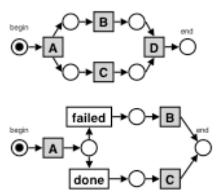
Visualisation of the Grid Job

Grid Job Description

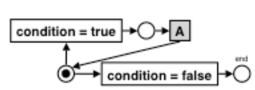
Jobhandler:
Runs the complete
Job sequence



Simple Petri Nets

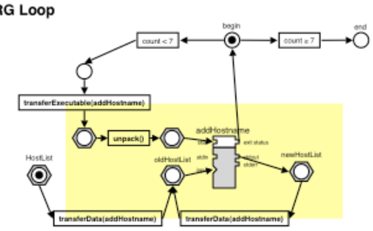


Petri nets: while(condition = true) do



Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

FhRG Loop

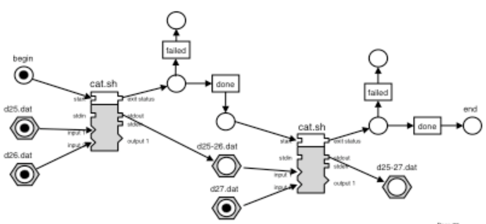


Page 34

Fraunhofer Institut für Arbeitswirtschaft und Organisation

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Workflow and Dataflow management with Petri Nets

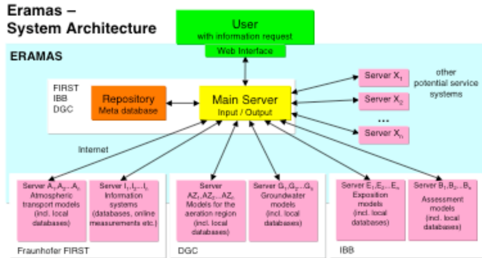


Page 35

Fraunhofer Institut für Arbeitswirtschaft und Organisation

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Eramas – System Architecture

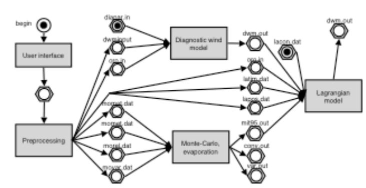


Page 36

Fraunhofer Institut Arbeitswirtschaft und Organisation

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Pollutant Transport in the Atmosphere

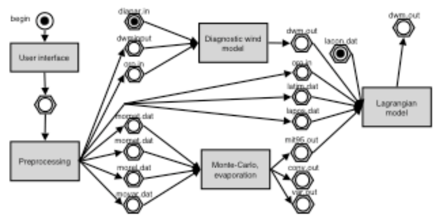


Page 37

Fraunhofer Institut Arbeitswirtschaft und Organisation

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Pollutant Transport in the Atmosphere



Page 37

Fraunhofer-Institute for Industrial Engineering /
Fraunhofer Resource Grid

Parts of our eXecution grid

- Grid Application Description Language
  incl. GJobDL based on Petri nets and able to handle very complex workflows
- Jobbuilder : user interface for creating grid job workflows
- Jobhandler : managing the execution
- resource broker
- web portal for secure access, task mapping ,...
- grid repository : Resource descriptions, User rights, ..
- based on Globus but independent of Globus
- easily portable to OGSI
- role based security

Page 38

Contact Information

http://www.fhrg.fhg.de

Juergen Falkner

Fraunhofer IAO

Nobelstr. 12

D-70569 Stuttgart, Germany

Tel : +49-(0)711-970-2414

Fax: +49-(0)711-970-2401

E-Mail: Juergen.Falkner@iao.fraunhofer.de

Dr. Franz-Josef Pfreundt
(FhRG Project Manager)

Fraunhofer ITWM

Europaallee 10

D-67657 Kaiserslautern, Germany

Tel : +49-(0)631-303-1800

Fax: +49-(0)631-303-1811

E-Mail: Pfreundt@itwm.fraunhofer.de

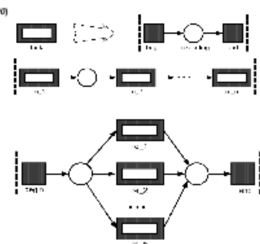
Petri Nets

(from van der Aalst und Kumar, 2000)

Task

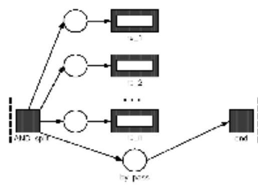
Sequence

Choice



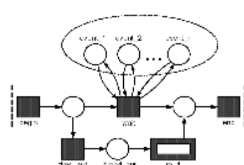
Petri Nets

Parallel execution without synchronization



Petri Nets

Wait all with time out



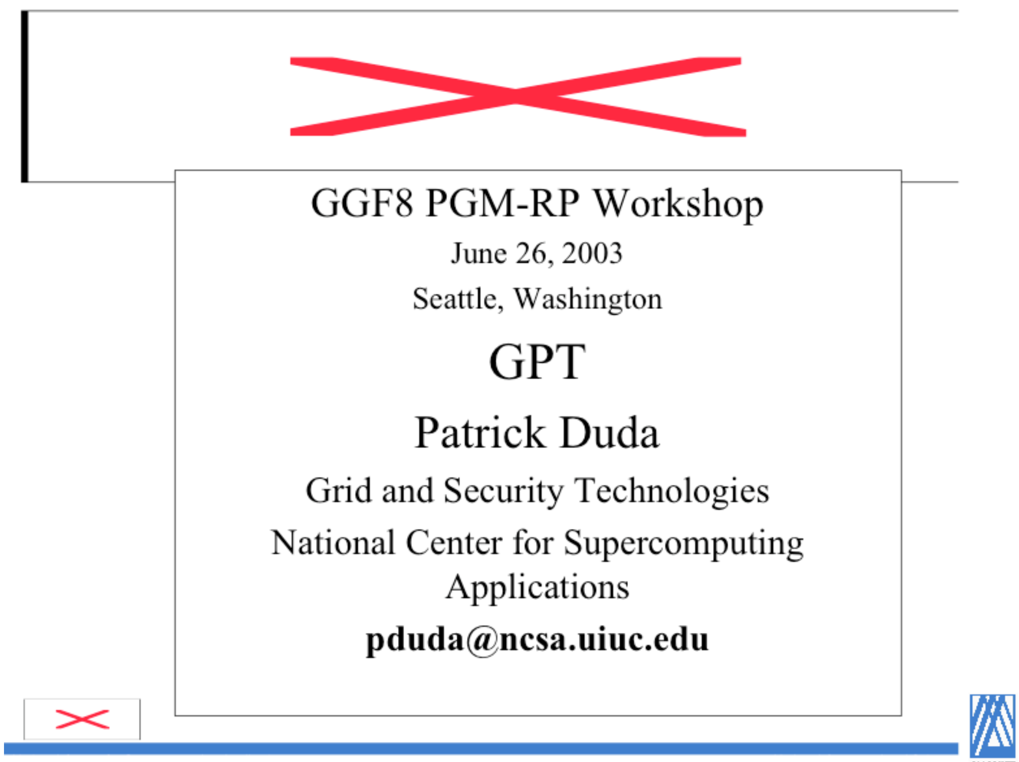
Grid Packaging Using GPT 3.x

Patrick Duda, National Center for Supercomputing Applications (NCSA)

Abstract

The newest version of the Grid Packaging Toolkit (GPT) has added enhancements to simplify the deployment of software onto a grid. Specifically, GPT bundles have been improved so that a set of packages can be managed in the same way individual packages are. GPT has also improved its diagnostic capabilities for the analyzing of existing installed software. These modifications allow for simpler maintenance of grid systems by allowing installed software to be queried and have only the needed updates take place. GPT bundling data can now be used to match a set of packages in an installation to bundle definitions that have been defined by a distribution organization such as NMI or Teragrid. GPT has also been extended to work with MDS and provide information about installed packages and bundles.

Presentation



- **Grid Packaging Tool**
  - What is GPT
  - Why GPT
  - Improved Features
  - New Features
  - What the Future Holds

"I don't see much sense in that," said Rabbit.

"No," said Pooh humbly, "there isn't. But there was going to be when I began it. It's just that something happened to it along the way."
(A.A. Milne)

ALLIANCE

- **A Tool For Deploying Grid Components**
  - A *(relatively)* easy way to package software
  - A means to building and installing collections of software packages
  - A set of standard version negotiation methods for managing updates to packages and bundles
  - A project funded by NMI
  - An open source effort

ALLIANCE

_ The Benefits Of Using GPT

_ Provides a consistent deployment mechanism across multiple systems/OSs/platforms

_ Both a developer tool and a deployment tool

_ Provides a reproducible software stack

_ Provides a site specific configuration capability

_ GPT vs. RPM

_ Relocatable binary packages

_ Originally RPM did not support

_ Current RPM supports but breaks often

_ GPT supports through "–location" flag

_ Multiple build cycles

_ RPM assumes single build cycle

_ Access to packaging data

_ RPM data is proprietary

_ GPT data is XML based

_ RPM is very good on Linux

_ GPT works on many platforms

_ GPT vs. RPM

_ Small footprint

_ RPM has numerous libraries(zlib, bzip2, 1 or 2 versions of the the berkeley database, and gettext)

_ GPT needs Perl 5.005 and zlib

_ GPT works better with Globus

_ Grid Packaging Tool

_ What is GPT

_ Why GPT

▪ **Improved Features**

_ New Features

_ What the Future Holds

_ GPT 3.x A Brave New World

_ Better Bundling

_ Better dependency checking

_ Better Updating-ability

_ Better Bundling

_ Bundle data XML based

_ Increases the amount of data about a bundle

_ Makes bundle data more accessible

_ Data is persistent

_ Install, Update, Delete

_ Simpler releases

_ Simpler Updates

_ Simpler system maintenance

gpt-bundle [options] packages

Options:	
-bn=NAME	Name of the bundle
-bv=MAJOR.MINOR	Version of the bundle
-bs=STABILITY	Stability of the bundle contents
-bl=VERSION	Version Label of the bundle
-template	Outputs empty Bundle Def XML file.
-bundledef=FILE	Outputs XML file with command line values.
-exclude=PACKAGE	Don't include PACKAGE in bundle
-xml=FILE	XML bundle description file
[packages]	List of packages to be bundled


```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE GPTBundleData SYSTEM "gpt_bundle.dtd">
<GPTBundleData Name="BUNDLE NAME" >
<BundleInfo >
<Description >Simple Description of the Bundle</Description>
<ContactInfo ContactEmail="Bundler's email address" ContactName="Bundler's name" />
<BundleDocs BundleDocsDesc="EMPTY" BundleDocsURL="http://www.gridpackagingtools.org" />
</BundleInfo>
<BundleReleaseInfo >
<BundleStability Release="Experimental" />
<BundleVersion Age="0" Major="1" Minor="0" />
<VersionLabel >2.0</VersionLabel>
<TypeOfBundle ContentsType="gpt" />
</BundleReleaseInfo>
<PackageList >
<IncludedPackages >
<Package PackageFlavor="gcc32" PackageName="globus_common" PackageType="pgm" PackageVersion="3.5" />
</IncludedPackages>
<ExcludedPackages >
<Package PackageFlavor="gcc32" PackageName="globus_common" PackageType="pgm" PackageVersion="3.5" />
</ExcludedPackages>
<PackageFlags >
</PackageFlags>
</PackageList>
</GPTBundleData>
```

Element or Attribute	Description	gpt-bundle Command Line Switch
GPTBundleData Name	Name of the bundle	-bn="NAME"
ContactInfo ContactEmail	Email address for questions about the bundle	None
ContactInfo ContactName	Name of the bundle owner	None
BundleDocs BundleDocsDesc	Explanation on how to get documentation on the bundle	None
BundleDocs BundleDocsURL	URL to online bundle documentation	None
BundleStability Release	Stability of the software released with this bundle. Usually one of "Experimental", "Alpha", "Beta", or "Production"	-bs="STABILITY"
BundleVersion Age, Major, and Minor	GPT internal version number of the bundle. Age is not currently used	-bv="MAJOR.MINOR"
VersionLabel	The user friendly version number of the bundle. Can be any CDATA string.	-bl="LABEL"

gpt-bundle -bn=newfoo -bv=1.0 -bl=2.0 -installdir=GLOBUS_LOCATION **–template** globus_ssl_utils-gcc32_pgm globus_common-gcc32_pgm

File: BundleTemplateXML_FILE.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE GPTBundleData SYSTEM "gpt_bundle.dtd">
<GPTBundleData Name="newfoo" >
<BundleInfo >
<Description >EMPTY</Description>
<PackagingTool ToolName="GPT" ToolVersion="3.0rc4_test" />
<ContactInfo ContactEmail="EMPTY" ContactName="EMPTY" />
<BundleDocs BundleDocsDesc="EMPTY" BundleDocsURL="EMPTY"
/>
<ComponentInformation ComponentName="NEED A NAME" >
<VersionLabel >NEED A VERSION</VersionLabel>
<Description >NEED A DESCRIPTION</Description>
</ComponentInformation>
</BundleInfo>
<BundleReleaseInfo >
<BundleStability Release="EMPTY" />
<BundleVersion Age="0" Major="1" Minor="0" />
<VersionLabel >2.0</VersionLabel>
<TypeOfBundle ContentsType="EMPTY" />
</BundleReleaseInfo>
```

```
<FileName >EMPTY</FileName>
<PackageList >
<IncludedPackages >
<Package PackageFlavor="gcc32"
PackageName="globus_ssl_utils" PackageType="pgm"
PackageVersion="0.0" />
<Package PackageFlavor="gcc32"
PackageName="globus_common" PackageType="pgm"
PackageVersion="0.0" />
</IncludedPackages>
<ExcludedPackages >
<Package PackageFlavor="PACKAGE FLAV"
PackageName="PACKAGE NAME" PackageType="ANY"
PackageVersion="0.0" />
</ExcludedPackages>
<PackageFlags >
</PackageFlags>
</PackageList>
</GPTBundleData>
```

```
gpt-bundle -installdir=GLOBUS_LOCATION -bn=newfee -bv=3.0 -bl=4.0 -bs=production -bundledef=tst2
globus_ssl_utils-gcc32_pgm globus_common-gcc32_pgm
```

File: tst2.gpt-bundle.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE GPTBundleData SYSTEM "gpt_bundle.dtd">
<GPTBundleData Name="newfee" >
<BundleInfo >
<Description >Test Description</Description>
<PackagingTool ToolName="GPT" ToolVersion="3.0rc4_test" />
<ContactInfo ContactEmail="Me@home.com" ContactName="Me" />
<BundleDocs BundleDocsDesc="Test Doc"
BundleDocsURL="www.ncsa.uiuc.edu" />
</BundleInfo>
<BundleReleaseInfo >
<BundleStability Release="production" />
<BundleVersion Age="0" Major="3" Minor="0" />
<VersionLabel >4.0</VersionLabel>
<TypeOfBundle ContentsType="gpt" />
</BundleReleaseInfo>
<FileName >EMPTY</FileName>
<PackageList >
<IncludedPackages >
<Package PackageFlavor="gcc32"
PackageName="globus_common" PackageType="pgm"
PackageVersion="2.0" />
<Package PackageFlavor="gcc32" PackageName="globus_common"
PackageType="rtl" PackageVersion="2.0" />
<Package PackageFlavor="gcc32" PackageName="globus_common"
PackageType="pgm" PackageVersion="2.0" />
<Package PackageFlavor="gcc32" PackageName="globus_common"
PackageType="rtl" PackageVersion="2.0" />
<Package PackageFlavor="noflavor"
PackageName="globus_common_setup" PackageType="pgm"
PackageVersion="2.0" />
<Package PackageFlavor="noflavor" PackageName="globus_core_setup"
PackageType="pgm" PackageVersion="2.0" />
<Package PackageFlavor="gcc32" PackageName="globus_openssl"
PackageType="pgm" PackageVersion="0.1" />
<Package PackageFlavor="gcc32" PackageName="globus_openssl"
PackageType="rtl" PackageVersion="0.1" />
<Package PackageFlavor="gcc32" PackageName="globus_ssl_utils"
PackageType="pgm" PackageVersion="2.1" />
<Package PackageFlavor="gcc32" PackageName="globus_ssl_utils"
PackageType="rtl" PackageVersion="2.1" />
<Package PackageFlavor="noflavor"
PackageName="globus_ssl_utils_setup" PackageType="pgm"
PackageVersion="2.0" />
</IncludedPackages>
<ExcludedPackages >
</ExcludedPackages>
<PackageFlags >
</PackageFlags>
</PackageList>
</GPTBundleData>
```

gpt-bundle switch	Bundle Definition File Element	Description
-nodeps	PackageFlags NoDeps	Do not include any dependent packages.
-nosetups	PackageFlags NoSetups	Do not include any dependent setup packages.
-exclude=PACKAGE NAME	ExcludedPackages	Do not include package that matches PACKAGE NAME.

gpt-bundle switch	Bundle Definition File Element	Description
-nodeps	PackageFlags NoDeps	Do not include any dependent packages.
-nosetups	PackageFlags NoSetups	Do not include any dependent setup packages.
-exclude=PACKAGE NAME	ExcludedPackages	Do not include package that matches PACKAGE NAME.

_ Installs

_ What should happen

bash$ gpt-install -verbose newfoo-2.2.3-i686-pc-linux-gnu-bin.tar.gz

globus_ssl_utils_setup-noflavor-pgm successfully removed.
globus_ssl_utils-gcc32-rtl successfully removed.
globus_ssl_utils-gcc32-pgm successfully removed.
globus_openssl-gcc32-rtl successfully removed.
globus_openssl-gcc32-pgm successfully removed.
globus_core_setup-noflavor-pgm successfully removed.
globus_common_setup-noflavor-pgm successfully removed.
globus_common-gcc32-rtl successfully removed.
globus_common-gcc32-pgm successfully removed.
Bundle newfoo removed.

globus_trusted_ca_42864e48_setup-noflavor-pgm successfully installed.
globus_proxy_utils-gcc32-pgm successfully installed.
globus_openssl_module-gcc32-rtl successfully installed.
globus_openssl-gcc32-rtl successfully installed.
globus_openssl-gcc32-pgm successfully installed.
globus_gsi_sysconfig-gcc32-rtl successfully installed.
globus_gsi_proxy_ssl-gcc32-rtl successfully installed.
globus_gsi_proxy_core-gcc32-rtl successfully installed.
globus_gsi_openssl_error-gcc32-rtl successfully installed.
globus_gsi_credential-gcc32-rtl successfully installed.
globus_gsi_cert_utils-gcc32-rtl successfully installed.
globus_gsi_cert_utils-gcc32-pgm successfully installed.
globus_gsi_callback-gcc32-rtl successfully installed.
globus_common_setup-noflavor-pgm successfully installed.
globus_common-gcc32-rtl successfully installed.
globus_common-gcc32-pgm successfully installed.
Bundle newfoo successfully installed.

_ Install Problems

_ What's more likely to happen

This example shows conflicts because of mismatched package versions:

bash$ gpt-install newfee-2.0-i686-pc-linux-gnu-bin.tar.gz

Error: The following package conflicts were found:
globus_openssl-gcc32-rtl ver: **0.1** in bundle newfee conflicts with globus_openssl-gcc32-rtl ver: **0.10** in bundle newfoo
globus_openssl-gcc32-pgm ver: **0.1** in bundle newfee conflicts with globus_openssl-gcc32-pgm ver: **0.10** in bundle newfoo
globus_common_setup-noflavor-pgm ver: **2.0** in bundle newfee conflicts with globus_common_setup-noflavor-pgm ver: **2.1** in bundle newfoo
globus_common-gcc32-rtl ver: **2.0** in bundle newfee conflicts with globus_common-gcc32-rtl ver: **3.5** in bundle newfoo
globus_common-gcc32-pgm ver: **2.0** in bundle newfee conflicts with globus_common-gcc32-pgm ver: **3.5** in bundle newfoo

_ More Install Problems

This example shows conflicts because of mismatched package flavors:

bash$ gpt-install newfee_static-2.2.3-i686-pc-linux-gnu-bin.tar.gz

Error: The following package conflicts were found:
globus_proxy_utils-gcc32-**pgm_static** ver: 0.5 in bundle newfee_static conflicts with globus_proxy_utils-gcc32-**pgm** ver: 0.5 in bundle newfoo
globus_openssl-gcc32-**pgm_static** ver: 0.10 in bundle newfee_static conflicts with globus_openssl-gcc32-**pgm** ver: 0.10 in bundle newfoo
globus_gsi_cert_utils-gcc32-**pgm_static** ver: 0.4 in bundle newfee_static conflicts with globus_gsi_cert_utils-gcc32-**pgm** ver: 0.4 in bundle newfoo
globus_common-gcc32-**pgm_static** ver: 3.5 in bundle newfee_static conflicts with globus_common-gcc32-**pgm** ver: 3.5 in bundle newfoo

_ Install Nirvana

_ Making all your install problems go away

A **-force** flag can be used to install bundles which cause conflicts:

bash$ gpt-install -force newfee_static-2.2.3-i686-pc-linux-gnu-bin.tar.gz

Bundle newfee_static successfully installed.

_ How Do I know It Really Worked

_ gpt-verify

gpt-verify [-bundle -help -version -man]

-bundle	Verify installed bundles only.
-help	Print a brief help message and exits.
-man	Prints the manual page and exits.
-version	Prints the version of GPT and exits.

bash$ gpt-verify

1st) Reports if there are packages missing from an installed bundle or if there are package version mis-matches.

Verifying Bundles...
Bundle: newfoo

2nd) Reports if there are any missing or version mis-matches with packages that are required by other packages.

Verifying run-time dependencies...

3rd) Reports if any setup packages which contain configuration files and scripts for other packages are missing.

Verifying setup dependencies...

Verifying setup packages...
The following setup packages still need to be configured via gpt-postinstall:
globus_trusted_ca_42864e48_setup-noflavor-pgm

ERROR: The collection of packages in /home/mbletzin/install/globus is not coherent!

The error comes from the fact that the /home/mbletzin/install/globus/setup/globus/setup-gsi script mentioned in the gpt-postinstall output had not been run.

- Better Updating-ability
  - GPT Does Better Checking
  - Package Updates
  - Bundle Updates

Update system with Packages.

gpt-install -location=<YOUR_PREFIX> newfoo-2.0-i686-pc-linux-gnu-bin.tar.gz

gpt-install -location=<YOUR_PREFIX> globus_common-3.5-i686-pc-linux-gnu-gcc32-pgm.tar.gz globus_common-3.5-i686-pc-linux-gnu-gcc32-rtl.tar.gz

gpt-verify shows version mis-matches for the bundle newfoo. It also shows dependency problems indicating that this was not a well designed update.

bash$ gpt-verify

Verifying Bundles...
Bundle: newfoo
Package globus_common-gcc32-rtl ver: 2.0 is a mis-match with the following installed packages
    globus_common-gcc32-rtl ver: 3.5
Package globus_common-gcc32-pgm ver: 2.0 is a mis-match with the following installed packages
    globus_common-gcc32-pgm ver: 3.5

Verifying run-time dependencies...
ERROR: The following packages are missing
Package Runtime-globus_common-ANY-pgm version 3.5 is incompatible with globus_common_setup-noflavor-pgm

Checks are made for conflicts between bundles and packages.

bash$ gpt-install newfoo-2.2.3-i686-pc-linux-gnu-bin.tar.gz

ERROR: The following package conflicts were found:
    globus_openssl-gcc32-rtl ver: 0.10 in bundle newfoo conflicts with globus_openssl-gcc32-rtl ver: 0.1 in bundle newfee
    globus_openssl-gcc32-pgm ver: 0.10 in bundle newfoo conflicts with globus_openssl-gcc32-pgm ver: 0.1 in bundle newfee
    globus_common_setup-noflavor-pgm ver: 2.1 in bundle newfoo conflicts with globus_common_setup-noflavor-pgm ver: 2.0 in bundle newfee
    globus_common-gcc32-rtl ver: 3.5 in bundle newfoo conflicts with globus_common-gcc32-rtl ver: 2.0 in bundle newfee
    globus_common-gcc32-pgm ver: 3.5 in bundle newfoo conflicts with globus_common-gcc32-pgm ver: 2.0 in bundle newfee

- Installs That Do Nothing
  - gpt-install -noaction

```
bash$ gpt-install -noaction newfoo-2.2.3-i686-pc-linux-gnu-
      bin.tar.gz

The following bundles would be removed
      newfoo ver: 2.0
The following bundles would be installed
      newfoo ver: 2.2.3
The following packages would be removed
      globus_ssl_utils_setup-noflavor-pgm ver: 2.0
      globus_ssl_utils-gcc32-rtl ver: 2.1
      globus_ssl_utils-gcc32-pgm ver: 2.1
      globus_openssl-gcc32-rtl ver: 0.1
      globus_openssl-gcc32-pgm ver: 0.1
      globus_core_setup-noflavor-pgm ver: 2.0
      globus_common_setup-noflavor-pgm ver: 2.0
      globus_common-gcc32-rtl ver: 2.0
      globus_common-gcc32-pgm ver: 2.0
The following packages would be installed
      globus_trusted_ca_42864e48_setup-noflavor-pgm ver:
0.5
      globus_proxy_utils-gcc32-pgm ver: 0.5
      globus_openssl_module-gcc32-rtl ver: 0.2
      globus_openssl-gcc32-rtl ver: 0.10
      globus_openssl-gcc32-pgm ver: 0.10
      globus_gsi_sysconfig-gcc32-rtl ver: 0.3
      globus_gsi_proxy_ssl-gcc32-rtl ver: 0.1
      globus_gsi_proxy_core-gcc32-rtl ver: 0.3
      globus_gsi_openssl_error-gcc32-rtl ver: 0.2
      globus_gsi_credential-gcc32-rtl ver: 0.5
      globus_gsi_cert_utils-gcc32-rtl ver: 0.4
      globus_gsi_cert_utils-gcc32-pgm ver: 0.4
      globus_gsi_callback-gcc32-rtl ver: 0.3
      globus_common_setup-noflavor-pgm ver: 2.1
      globus_common-gcc32-rtl ver: 3.5
      globus_common-gcc32-pgm ver: 3.5
```

gpt-query [-name=Name -flavor=Flavor -pkgtype=Type --help -file=name -man -deps=run|sdk] [pkgname-flavor-pkgtype+]

-name=NAME	Returns all of the packages matching NAME.
-flavor=FLAVOR	Returns all of the packages matching FLAVOR.
-pkgtype=TYPE	Returns all of the packages matching TYPE.
-file=FILE	Returns the package owning the FILE. FILE needs to be either an absolute path or referenced to $GLOBUS_LOCATION.
-deps=run|sdk	Returns the runtime or build dependencies of a package.
-help	Print a brief help message and exits.
-man	Prints the manual page and exits.
-version	Prints the version of GPT and exits.
-what-bundles	Prints a list of the installed bundles.
-verify-bundle=BUNDLE	Takes a bundle defintion and checks to see if it matches the installation.
-**bundle-packages=BUNDLE**	List the packages that belong to the desired bundle.
-**bundle BUNDLE 1 [BUNDLE 2]**	If two bundles are given, the bundles will be compared to one another for equality. If only one bundle is given, the bundle will be checked against the installation. This command works on tar bundles.

gpt-query -location=/home/pduda/tmp2/inst -verify-**bundle=fee-2**

10 packages were found in /home/pduda/tmp2/inst that matched your query:

packages found that matched your query
globus_common-gcc32-pgm pkg version: 2.0.0
globus_common-gcc32-rtl pkg version: 2.0.0
globus_common_setup-noflavor-pgm pkg version: 2.0.0
globus_core_setup-noflavor-pgm pkg version: 2.0.0
globus_gridftp_server-gcc32-pgm pkg version: 1.0.0 software version: 1.0
globus_openssl-gcc32-pgm pkg version: 0.1.1
globus_openssl-gcc32-rtl pkg version: 0.1.1
globus_ssl_utils-gcc32-pgm pkg version: 2.1.0
globus_ssl_utils-gcc32-rtl pkg version: 2.1.0
globus_ssl_utils_setup-noflavor-pgm pkg version: 2.0.0

ALLIANCE

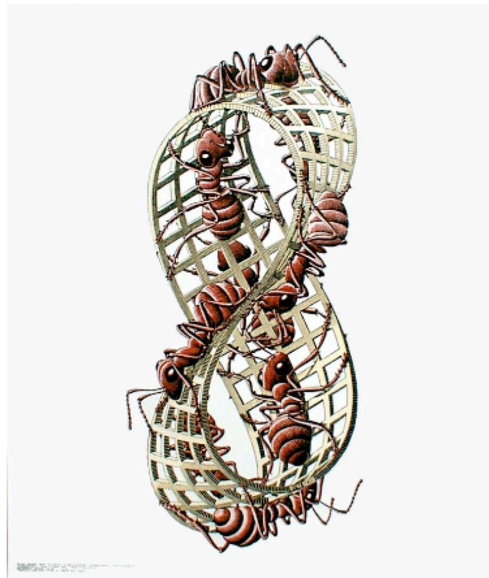
gpt-query -location=/home/pduda/tmp2/inst **-bundle-packages=fee-2**

Packages in Bundle fee-2:
globus_common-gcc32_rtl
globus_core_setup-noflavor_pgm
globus_gridftp_server-gcc32_pgm
globus_openssl-gcc32_pgm
globus_openssl-gcc32_rtl
globus_ssl_utils_setup-noflavor_pgm
globus_common-gcc32_pgm
globus_ssl_utils-gcc32_rtl
globus_common_setup-noflavor_pgm
globus_ssl_utils-gcc32_pgm

ALLIANCE

gpt-query -location=/home/pduda/tmp2/inst **-what-bundles**

System Bundles
    fee-2.0 ver: NONE



gpt-query -location=/home/pduda/tmp2/inst **-bundle fee-2.0-i686-pc-linux-gnu-bin.tar.gz fee-2.2.3-i686-pc-linux-gnu-bin.tar.gz**

fee-2.2.3- is missing:
    globus_core_setup-noflavor-pgm-2.0.0
    globus_ssl_utils_setup-noflavor-pgm-2.0.0
    globus_ssl_utils-gcc32-rtl-2.1.0
    globus_ssl_utils-gcc32-pgm-2.1.0

Bundle package defferences:
    fee-2.2.3 -: globus_common-gcc32-2.0.0
    fee-2.2.3 -: globus_common-gcc32-3.5.0
    fee-2.2.3 -: globus_gridftp_server-gcc32-1.0.0
    fee-2.2.3 -: globus_gridftp_server-gcc32-1.5.0
    fee-2.2.3 -: globus_openssl-gcc32-0.1.1
    fee-2.2.3 -: globus_openssl-gcc32-0.10.0
    fee-2.2.3 -: globus_openssl-gcc32-0.1.1
    fee-2.2.3 -: globus_openssl-gcc32-0.10.0
    fee-2.2.3 -: globus_common-gcc32-2.0.0
    fee-2.2.3 -: globus_common-gcc32-3.5.0
    fee-2.2.3 -: globus_common_setup-noflavor-2.0.0
    fee-2.2.3 -: globus_common_setup-noflavor-2.1.0

_ GPT MDS Information Provider

- _ MDS provides a standard mechanism for reporting information about a grid
- _ GPT-IP supplies data on installed packages and bundles
  - _ Package Name, Type, Version, and Flavor
  - _ Bundle Name, Version, and member Packages
- _ Open to extension

_ Improved, Better, Enhanced, Useful, Semi-understandable Documentation

- _ Comprehensive examples
- _ Insights into the inter-workings
- _ Step-by-step walk throughs

_ Where Can All This Be Found?

- _ http://www.gridpackagingtools.org
- _ http://www.gridpackagingtools.org/book.html

_ **GPT Repository Tool**
  - _ Think apt-get
  - _ Ties to GridConfig

_ **Generalized Query Tool**

_ **Grid Wide Package Information**
  - _ Information about multiple installed packages

_ **Improved MDS Information**

_ **Simpler Interface**

_ **Support For Globus 3.x**
  - _ Java builds
  - _ Packaging of Databases(postgres, mysql,…)
  - _ Web Servers(apache,…)
  - _ And more…