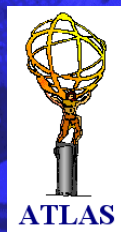


Grids, Data Grids, and High Energy Physics...

A Melbourne EPP perspective.

Lyle Winton

winton@physics.unimelb.edu.au



Overview



- ◆ Grids <<<<<
- ◆ Data Grids <<<<<
- ◆ Data Grids in practice <<<<<
- ◆ Overview of products
- ◆ How it works
- ◆ HEP Projects
- ◆ ATLAS Data Challenges
- ◆ University of Melbourne
- ◆ Belle Experiment

Before starting I'll give an overview of this presentation.

I'll talk about grids and what they are, in particular Data Grids as these are of interest to high energy physics. I'll discuss what grids are in practice and give an overview of the products that are available and how they work.

We'll look at grid projects within the field of High Energy Physics taking a closer look at LHC and ATLAS. And finally I'll talk about what we are doing at Melbourne University and focus on 2 major projects.

But first we'll look at Grids and Data Grids.

Grids



- ◆ “The Grid refers to an infrastructure that enables the integrated, collaborative use of high-end computers, networks, databases, and scientific instruments owned and managed by multiple organizations.”
- The **Globus Project**
- ◆ “Tomorrow, the grid will be a single, sustained engine for scientific invention. It will link petaflops of computing power, petabytes of data, simulation and modeling codes of every stripe, sensors and instruments around the globe, and tools for discovering and managing these resources. At your desktop and at your whim, you'll have access to the world and its computing assets.”
- **NCSA** (National Center for Supercomputing Applications)
- ◆ The grid “...consists of physical resources (computers/clusters, disks and networks) and "middleware" (software) that ensures the access and the co-ordinated use of such resources.”
- **EDG** (European Data Grid)

We have a definition provided by Globus the organisation that provides the base for most grid software. The reason why we are interested in the Grid is that this describes a lot of what we do in experimental high energy physics.

The Organisation NCSA, a US based grid pioneer, outlines a vision for the GRID. While this may seem a bit utopian, the linking of much computing power and data is something that is required now, and the need will increase in the future.

Finally, the European Data Grid, who work closely with CERN, have outlined what the grid will be constructed of. It will consist of physical resources, sometimes termed “fabric”, and the middleware or software that joins these together.

Grids



- ◆ The Grid is not “The Internet 2”
 - it uses existing internet infrastructure
 - however, Grid projects exist to extend current infrastructure
- ◆ The Grid does not replace Cluster or Parallel computing.
 - it does not fix the problems of SMP/clustering
 - it is designed to help manage, share, and utilise existing technologies and resources
 - clusters / parallel machines becomes Grid nodes
 - » resource information and status are available to Grid users
 - » resources may be utilised by Grid users
- ◆ The Grid does not make resources bigger or faster
 - may provide easier access to more resources

A common misconception is that the Grid is the internet mark 2. It is not. It uses the existing internet infrastructure. However, there are some grid projects aimed at extending the existing internet infrastructure.

The grid is not designed to replace Cluster or Parallel computing. It does not fix the problems and limitations with technologies such as SMP or clustering. In fact, it is designed to help us utilise these technologies, and share computing resources.

In the grid a cluster or SMP machine becomes a single Grid Node or Computation Resource. Detailed information about these resources and their status are made available to the Grid, and jobs may be submitted to these nodes/resources based on this information and your requirements.

Grids



- ◆ Analogy - a power grid
 - Nodes that supply power and nodes that use power
 - Standard connections are required (eg. 240V 50Hz)
 - Users access and consume power as required transparently.
 - Easy as switching on a light! (?)

Another way to describe the grid is with an analogy. We can say that The Grid is like an electrical power grid.

We have nodes that supply power (which are resources that provide computing power) and nodes that use power (users accessing the grid).

Standards are required to connect these together seamlessly. (Users and computing facilities must be using the same standards or applications.)

Users will access and consume power (in our case computing resources) as required without needing to know details about where the resources are coming from.

Data Grids



- ◆ What?
 - Grids where access to data is as important as access to computational resources
- ◆ Where?
 - Collaborative research or data intensive research...
 - Experimental High Energy Physics (HEP)
 - » data focused, simulation & analysis data, large collaborations
 - » LHC from 2006-2007 will produce 8 PB/yr raw data (8×10^{15} bytes ; not all required by all users)
 - » accessible to 1000's of users in many institutes
 - Earth Systems Grid (ESG)
 - » climate studies in the US
 - » 3 PB/yr data requiring 3 TFLOPS processing power (2005)
(1 MFLOP \approx P4 @ 1 MHz ; 3 TFLOP \approx 1500 P4 2GHz)

Data Grids. What are they? These are grids where the access to the data is as important, or more important, than access to computational resource.

These occur in most collaborative research areas where access to common data is required or data intensive research areas. High Energy Physics is a good example as it is a data focused research area where access to simulation and analysis data is required by large collaborations of institutes spread around the world.

To give an indication LHC will produce around 8PB of raw data per year. Now, only a small fraction of this will be needed by any one user but this data must be accessible to 1000's of users world-wide.

Another example is the Earth Systems Grid in the united states. They will produce 3 PB of data per year which will require 3 TFLOPS of processing power to analyse.

Data Grids in Practice



Current Concerns:

- ◆ Administration
 - time on system administration
 - required software
 - accounts, security, network...
- ◆ Data Management
 - store/Xfer of large data is \$\$\$
 - backup is \$\$\$/impossible
 - software changes X many users = proliferation of data
- ◆ Computational Resources
 - CPU required for peak loads \$\$\$
 - idle/wasted CPU if usage varies
- ◆ Service Replication
 - reinventing of code for...
 - » distributed/parallel computing
 - » data or database access
 - » access to mass storage
 - » packaging of software

Data Grid Goals:

- ◆ Administration:
 - Simple grid-wide admin tools (security/data/software)
- ◆ Data Management:
 - world-wide access to data
 - intelligent data storage, caching, replication (faster & cheaper)
 - better description & versioning
- ◆ Computational Resources:
 - wider access to more CPU
 - spread of load may lead to less idle time (less CPU required)
- ◆ Services:
 - intelligent job/process location (automatic move code/data ; faster process time ; reduced cost)
 - transparent network usage
 - standard API
 - software packaging tools

Why bother with using grids? Haven't we been managing with existing methods?

In short, it's going to get harder because collaborations, experiments, and the data are all getting larger. The list of concerns is long.

From an administration point of view much time is already spent. Looking at data management, the storage and transfer of large amounts of data can be expensive. Commercially data transfer can cost of the order of cents/MB, disk storage around a cent/MB, and tape storage around a dollar/GB. Backup of this data, particularly using traditional methods, can be expensive or impossible. Changes in the software or different software together with many users and their private analyses can lead to the proliferation of data. This all needs to be tracked and managed.

From the point of view of computing power, the CPU required for peak times can be very expensive, however idle CPU exists if loads vary.

Service or Software replication exists between high energy physics experiments. Users continue reinventing code to perform common services or tasks to meet the needs of their experiments.

So the advantages, or more accurately goals, of data grids are mainly solutions to these:

Providing simple centralised administration tools.

Providing data tools for world-wide access. Intelligent data storage, caching, and data replication for faster and cheaper access. The grid may also provide a better way of filing data with well formed descriptions and version information. These features could be of the most benefit to high energy physics.

The grid will provide access to geographically separated computing facilities. This could mean access to more computing power. The spreading of CPU load may mean that less CPU is required for each facility to meet peak loads.

The grid will provide standard software services and APIs such as job management. Intelligent job locating like moving the code to where the data is or vice-versa for faster processing times and reduced cost. To some extent we currently do this by hand. Standard APIs will be provided for parallel computing, network access, and for many other common tasks.



◆ Problems?

- How are we going to provide the above?
- grids of 10,000+ computers must be administered
- user groups of 1000+ must be administered
- maintain access privileges for users/data/space/computers
- monitoring of distributed processes, data, resources
- job profiling and resource profiling (what job can run where)
- job management (where and when will the job run)
 - » CPU time / CPU count (for MPI/PVM)
 - » other resources (memory, disk scratch...)
 - » data location
 - » current resource loads or availability
 - » time/cost of data transfer
 - » cost of CPU usage (smaller institutes may rent CPU time)
 - » user/job permissions and priorities
 - » resource restrictions/permissions and priorities

Most of the above goals are easier said than done! How do we do it?

We have to provide for administration of 10's of thousands of computers, thousands of users, access privileges for all of this and the data we produce, monitoring facilities, descriptions of user jobs and the computers (so we know what can run where), job management which will determine where and when the job will run using information such as require CPU time or number of CPUs for parallel jobs, required memory or hardware, location of the data, current loads on resources, the time or cost of data transfer, perhaps the cost of CPU (as smaller institutes may choose to rent CPU from larger organisations), user permissions, and resource restrictions.

A lot of these problems have not been solved and a lot of software packages are still in the development and testing phases.

Overview

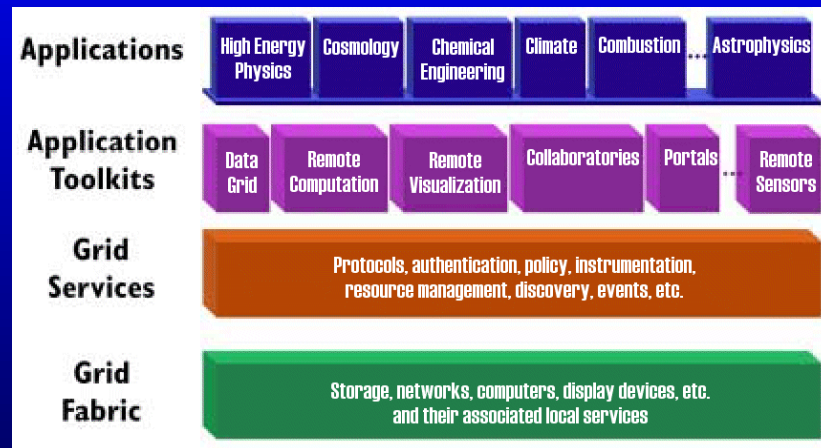


- ◆ Grids
- ◆ Data Grids
- ◆ Data Grids in practice
- ◆ Overview of products <<<<<
- ◆ How it works
- ◆ HEP Projects
- ◆ ATLAS Data Challenges
- ◆ University of Melbourne
- ◆ Belle Experiment

Overview of Products



- ◆ Data Grid services and software are termed “middleware” (they lie between the fabric/systems and the users/applications)



Before starting on an overview of products you may recall I mentioned the term middleware is used to describe grid services and software. This diagram tries to illustrate the relationship between the various parts within a grid.

The grid services sit on top of the hardware (or fabric) and provide the means of communication with the hardware including security.

The application toolkits, of which data grid specific services are a part, provide tools and interfaces for higher level grid usage, such as job scheduling and data management.

The applications themselves we are all familiar with. They are the tools and applications that we are currently running.

Much effort has gone into the development of the grid services and the application user groups have also contributed. The application toolkits are probably less advanced and it's harder to predict which may emerge as leading future toolkits. Because of this, we've concentrated mainly on the grid services and bridged the gap by either developing or using simple application tools where absolutely necessary.

Overview of Products



- ◆ Available Packages
 - Globus Toolkit
 - » the core of most grid middleware
 - European Data Grid (EDG, or EU-DataGrid or DataGrid)
 - Virtual Data Toolkit (VDT - GriPhyN project)
 - Particle Physics Data Grid (PPDG)
 - » no package as yet, just a collections of tools
 - Nile (CLEO experiment)
 - » Java, CORBA/OrbixWeb based
 - » not built on Globus!!!
 - Sun™ Grid Engine (SGE), Enterprise Edition
 - more?

Looking at some of the main products that are available we have:

The Globus Toolkit which is at the core of most grid middleware. (see previous slide) This provides the low level grid services.

EDG, VDT, PPDG are examples of data grid packages that provide the higher level services. These are all built on the Globus Toolkit.

Nile and Sun Grid Engine are a notable products they appear to be the only ones which are not built on a Globus core.

Many of these packages can be broken down into separate services or products.

Overview of Products



◆ Available Tools and Services

(Grid Acronym Soup! GAS)

- **Security**
 - GSI - Grid Security Infrastructure (globus)
- **Service/Resource Information (data/machine)**
 - MDS - Metacomputing Directory Service (globus)
 - GRIS - Grid Resource Information Service (globus)
 - GHS - Grid Index Information Service (globus)
- **Resource Management**
 - RSL - Resource Specification Language (globus)
 - method to exchange resource info & requirements
 - GRAM - Globus Resource Allocation Manager (globus)
 - standard interface to computation resources like PBS/Condor
 - DUROC - Dynamically-Updated Request Online Coallocator (globus)
 - WMS - Workload Management System (EDG)
- **Data Management**
 - GSIFTP - high performance, secure FTP uses GSI (globus)
 - Replica Catalog - data filing and tracking system (globus)
 - GASS - Globus Access to Secondary Storage (globus)
 - access data stored in any remote file system by URL
 - Unix like calls fopen(), fclose()
 - GDMP - Grid Data Mirroring Package (EDG,GriPhyn,PPDG)

So here is a list of available products. And this is not an exhaustive list!

You need only know that the basis for most grid environments is the Globus Toolkit.

Another thing you can tell from this list is that the grid, being a computing concept, is plagued with acronyms. This is sometimes called the Grid Acronym Soup (GAS). There are large web pages dedicated to defining all of these acronyms. (<http://www.gridpp.ac.uk/docs/GAS.html>)

Some of the central services are GSI which provides the security, GRIS which provides the information about grid resources, GRAM which runs tasks on computing resources, GASS and GSIFTP which allows access to remote files, and the Replica Catalog which provides the filing system for the data.

Overview of Products



◆ Available Tools and Services (cont.)

- **Data Management (cont.)**
 - Magda - distributed data manager (PPDG)
 - Spitfire - grid enabled access to any RDBMS (EDG)
 - RLS - Replica Location Service (EDG)
- **Mass Storage**
 - HPSS API - high performance storage system (globus)
 - SRB API - Storage Resource Broker (globus)
- **Communication**
 - Nexus and MPICH-G (globus)
- **Monitoring**
 - HBM - Heartbeat Monitor (globus)
 - many others!
- **Job Managers**
 - PBS - portable batch system
 - Condor - distributed computing environment
- **Fabric Management**
 - Cfengine
 - PacMan
 - many others!

And the list continues.

Overview



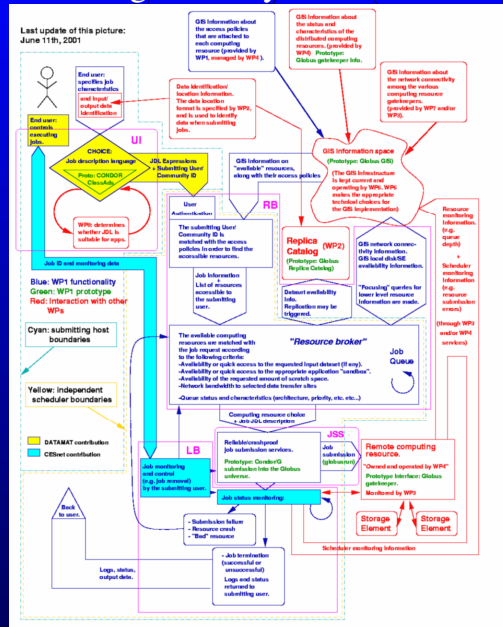
- ◆ Grids
- ◆ Data Grids
- ◆ Data Grids in practice
- ◆ Overview of products
- ◆ **How it works** <<<<<
- ◆ HEP Projects
- ◆ ATLAS Data Challenges
- ◆ University of Melbourne
- ◆ Belle Experiment

Now we will look at how some of these products work, or more appropriately, how the grid is supposed to work, as many products are still in development.

How it's supposed to work!



◆ Workload Management System (EDG example)



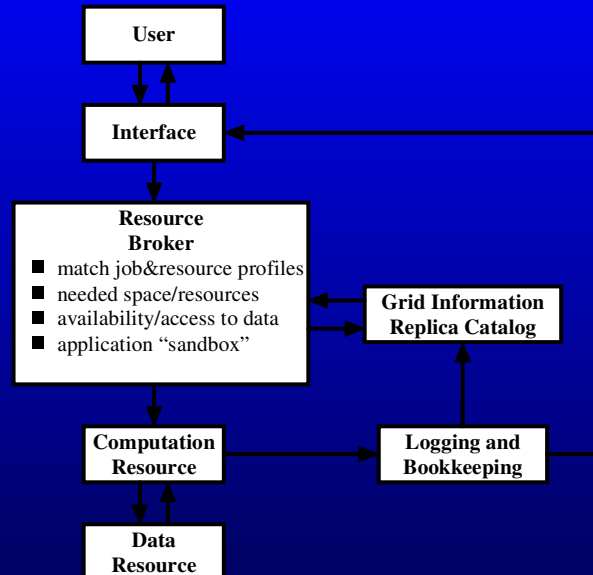
This is the official flowchart for the workload manager in the European Data Grid package. This simply takes a job and executes it on a computing resource. But as you can see the process isn't that simple.

Because this would take a while to explain I've simplified the process.

How it's supposed to work!



◆ Workload Management System (EDG; very simple version)



In the simplified the process we can get an idea of what's happening.

A user submits a job via some sort of interface. This is passed on to the resource broker which does the majority of the work.

It is the Resource Broker's task to match the Job Profile with the Resource Profiles that are available. It checks how much disk space, memory, cpu, and network bandwidth you need to run the job. It checks the location, availability, and access permissions to the required data. This is done by accessing the grid information services and replica catalogue. It ensures the presence of what's called the "application sandbox" which may include application code and all necessary auxiliary files. If it needs to it can move the data to an appropriate location before the job starts and updates the replica catalog accordingly. It then actually runs the job on one or more computers.

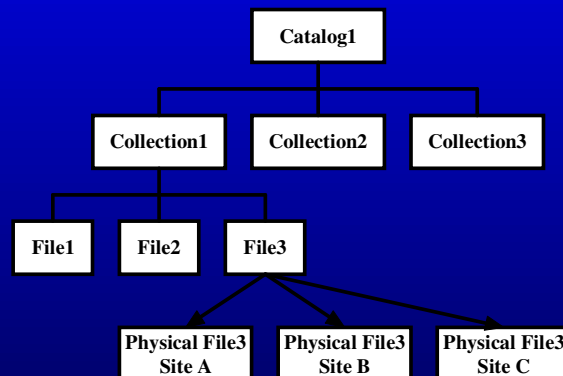
The status and output of the job are monitored so the grid information can be kept up to date and so the user knows what's happening.

How it's supposed to work!



◆ Replica Catalog

- Data can be organised as “Logical File Names” in a virtual directory structure which can be mapped to “Physics File Names”
- Logical file structure is organised into Catalogs, Collections, Files



As we are dealing with Data Grids and data is important I thought we should also look at file management and replication within the grid.

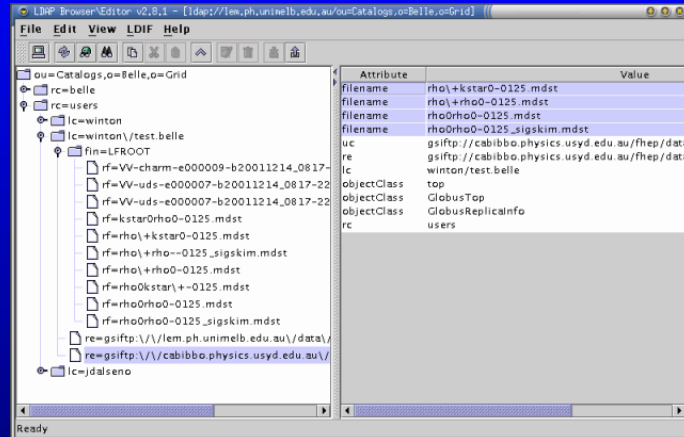
Data files can be organised as Logical File Names within a Virtual Directory Structure by using a Replica Catalog. Each Logical File Names maps to one or many Physical Files located somewhere on grid, usually specified as URLs. The virtual directory structure is organised into Catalogs and Collections.

In a typical job the user will specify the Logical File that is to be processed and the Resource Broker will resolve this to the most appropriate Physical File location. The job may require the file to be staged locally before execution or the application will stream the file from it's physical location.

How it's supposed to work!



◆ Replica Catalog - LDAPBrowser



- ◆ “Meta-Data” - easily added to LDAP directory
 - data descriptions and version information
 - LDAP query language can retrieve files by description/version

The Replica Catalog is generally implemented using an LDAP directory. The advantage is that any existing LDAP tool can be used, such as the one displayed, and LDAP queries can be used to obtain information.

The displayed tool, called LDAPBrowser, is shown examining logical files within a collection “winton/test.belle” which is found within the “users” catalog. In this type of Replica Catalog physical files are determined by location entries, such as the one highlighted, which are linked to the logical files.

An LDAP directory will also allow the adding of extra attributes to entries which may be used to store “meta-data”, extra information attributed to the data. This could be used to store well formatted descriptions of the data and version information. The LDAP query language can then be used to retrieve data by description or version.

How it's supposed to work!



- ◆ Replica Catalog - Grid-RC-tools (U.of.M, Physics)
 - developed to emulate Unix directory structure commands

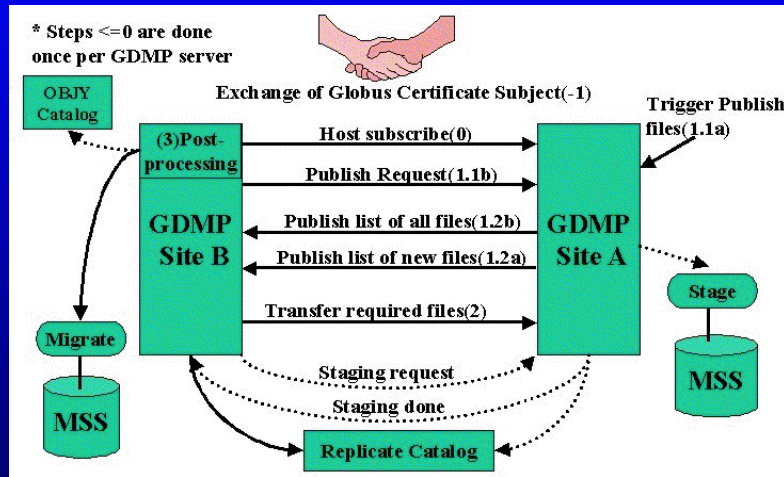
```
> grid-rc-cd winton/mcset1
> grid-rc-mkdir newcollection
RC Password: *****
> grid-rc-ls -l
drwxr-x Lyle_Winton      2002-11-18_03:36          0  .
-rw-r-- Lyle_Winton      2002-11-18_03:35      503589128  myfile3.mdst
-rw-r-- Lyle_Winton      2002-11-18_03:35      516000000  myfile4.mdst
-rw-r-- Lyle_Winton      2002-11-18_03:35      167506804  myfile5.mdst
> grid-rc-cp -local myfile1.mdst . gsiftp://remote/dir/
> grid-rc-cp gsiftp://remote2/dir/ myfile2.mdst
> grid-rc-cp myfile2.mdst gsiftp://remote3/adir/
> grid-rc-rm myfile3.mdst
> grid-rc-location '*.mdst'
/users/winton/mcset1/myfile1.mdst: gsiftp://remote/dir/myfile1.mdst
/users/winton/mcset1/myfile2.mdst: gsiftp://remote2/dir/myfile2.mdst
                                gsiftp://remote3/adir/myfile2.mdst
/users/winton/mcset1/myfile4.mdst:
/users/winton/mcset1/myfile5.mdst: http://somehost/otherdir/myfile2.mdst
> grid-rc-setattr description="MC D*D*Ks" 'myfile?.mdst'
> grid-rc-find -r /users/winton(size>=1000)
```

We have developed a set of tools for querying and maintaining the replica catalog. These tools emulate the Unix directory commands so the user can navigate the virtual directory structure (replica catalog) as they would normally navigate their directories. A few example commands are displayed here. The “CP” or “copy” command not only copies files around the replica catalog but also provides a means for replicating files across grid data resources. The “location” command is used to report the locations of files. The “setattr” command allows the setting of extra attributes on files and collections. This will provide the foundation for meta-data maintenance. The “find” command is used to retrieve data or files from any point in the directory structure that match an LDAP query string.

How it's supposed to work!



◆ File Replication (GDMP example)



The GDMP tool is used replicates files from one system to another.

The diagram show a transfer of files from site A to site B. This can be triggered in two ways, by site A publishing a list of files that need to go to other sites (1.1a), or by site B requesting a list of files (1.1b). In the later case, Site B may be configured to request certain files that will be used commonly, or a user application running at site B may request files for processing.

Once the file has been transferred, the Replica Catalog, which holds the list of physics file locations, is updated so other applications know to get this file from site B also.

How it's supposed to work!



◆ Object Replication vs File Replication

- Existing experimental data (file replication)
 - » files of many independent events (compressed, random access is difficult or impractical)
 - » groups of files of similar events are called “data-sets”
 - » large data-sets filtered for most interesting events, stored as “skims”
- Object replication seems the most efficient
 - » each event is an object potentially stored separately
 - » a data-set or skim is just a collection of unique event IDs
 - » no event duplication in multiple data-sets or skims
 - » data processing accesses the nearest event
- However, many current storage systems have scalability problems (events $> 10^9$ in number)
 - » file replication problems solved (in part) by replication of smaller common skims
 - » Compromise?: skim files (re)constructed by extracting events from the nearest data-set OR events in data-sets are indexed

A subject of note within the grid community is that of Object Replication.

Existing experimental HEP data is stored and replicated as files containing many independent events. Files and groups of files generally contain similar types of events that are typically processed together, and these are called data-sets. These data-sets are generally filtered to extract events of most interest into smaller data-sets called skims.

The computing community in general is moving towards treating data as objects rather than streams of information. At first this seems the most efficient way of dealing with data. In HEP each event is already an independent object which could be stored separately. Data-sets would become a collection of event IDs. Skims, for example, would just contain references to events and would be very small. As events in data-sets and skims may overlap with events in other data-sets, the duplication of events will be eliminated, because only event references are stored. Data processing will be performed on the nearest event. Some events may be cached locally, some in Sydney, some in CERN. Whereas with file replication, generally processing will occur only in places that have the whole file or the whole file must be transferred.

The problem with object replication is most current storage systems cannot handle the quantity of objects required, which may be greater than 10^9 in number. Most databases and file systems are not that powerful. So this problem is currently solve by having smaller skims shared by many people rather than large data sets and replicating these.

There may be a compromise where skim files can be constructed or reconstructed at process time by extracting events from the nearest data file or skim. Another way might be to index event locations within the large data-sets. But this still requires the tracking of large numbers of events.

How it's supposed to work!



◆ A User Example

– Traditional method

- » `ssh remote qstat` (choose location to run)
- » `scp files remote:dir` (transfer auxiliary files:
user code/libs, scripts, config)
- » `ssh remote qsub < myjob.csh` (run job or submit to queue)

– Globus method (Grid resources)

- » `grid-proxy-init` (security sign on)
- » `grid-info-search remote` (choose Grid node to run)
- » `globus-url-copy files remote` (transfer auxiliary files)
- » `globus-job-run myjob.csh` (submit job to node)

◆ Advantages:

- ◆ Authenticate once, run anywhere (without agents)
- ◆ Greater access to resources (when more exist)
- ◆ Access to remote data resources

◆ Disadvantages:

- ◆ Essentially the same as the traditional method

Here is an example of how a user will see the Grid system.

We will start with the traditional method, using SSH to connect to remote resources. A user must first choose a location to run their job, then copy their auxiliary files such as libraries, scripts, and configuration files to the remote site, then either run the job or submit it to an existing queue or batch system. This is repeated for as many job as you need to run.

With Globus installed we may access grid computation resources. A user signs on to the grid by creating a proxy, they query grid resources to choose an appropriate location to run, then copy auxiliary files, then submit the job to the resource. This has the advantage of a single point of authentication and potentially greater access to remote resources. The big disadvantage is that this is essentially the same as the traditional method. This is not too surprising as Globus only provides low level service access to the grid. The real benefit will be seen when using application tools as seen in the following examples.

How it's supposed to work!



◆ A User Example (cont.)

– Nimrod/G method (Monash, David Abramson; economic scheduler which accesses multiple resources; under development University Melbourne GridBus www.gridbus.org)

```
» grid-proxy-init  
» nimrod myjob.run  
» <Specify budget/deadline>
```

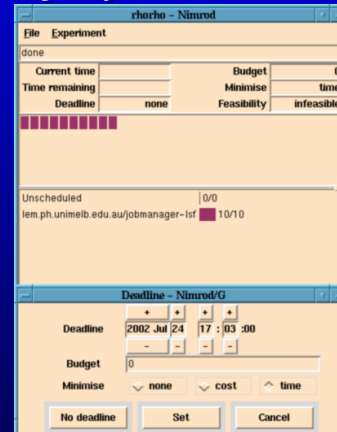
(security sign on)
(register jobs and start Nimrod)

◆ Advantages:

- ◆ File transfer is handled
- ◆ Grid nodes choice is handled, multiple can be used at once
- ◆ Cost and time considerations including feasibility

◆ Disadvantages:

- ◆ Cost of data transfer and data location is not yet considered (in development)



We are also trialing at Melbourne University the use of an economic scheduler called Nimrod as an interface to grid resources. An example of using this is to sign-on to the grid by creating a proxy, then to register and start a predefined plan for a group of jobs. You can then specify a budget or deadline for this run. The advantages are that file transfer, the choice of grid resource, and cost and time considerations are all handled by Nimrod. A single run can easily be split over multiple resources. The disadvantage is that data location and cost of transfer is not built into the scheduler, but we are working with the University of Melbourne GridBus group to change this.

How it's supposed to work!



◆ A User Example (cont.)

– Resource Broker method (EDG Workload Manager)

» `dg-job-submit myjob.profile` (security sign on & submit)

◆ Advantages:

- ◆ Very simple!
- ◆ File transfer is handled by Resource Broker
- ◆ Grid nodes choice is handled by Resource Broker including complex requirements, multiple nodes can be used

◆ Disadvantages:

- ◆ Cost, time, and feasibility are not considered (yet?)

– Ideal future method

- » Simple usage, authenticate once
- » File transfer and access handled transparently
- » Choice of Grid resource handled transparently including complex requirements, multiple nodes can be used at once
- » Cost/quota, time, feasibility consideration, particularly data access

And finally, I'll use an example of what how using a Resource Broker will look to the user. This example is the European Data Grid Workload Manager.

A job profile is simply submitted to the Resource Broker and security sign-on and everything else is handled in the one place. The advantages are this is very simple, file transfer and choice of resource is handled by the broker. The disadvantage is that cost, time, and feasibility are not yet considered by resource brokers.

So the ideal future method for submitting jobs to the grid should be simple to use, handle all file transferring and access, handle the choice of grid resource taking into account the requirements of the job, and also take into account cost, quotas, time, and job feasibility with respect to computing resources and data resources. Within data grids ensuring access to data resources is of particular importance.

How it's supposed to work!



◆ A User Example (cont.)

– We have developed a tool to meet our most important requirement - data access

– GQSched method (Grid Quick&Dirty Scheduler)

» `gqsched gatekeepers myjob.csh` (security sign on & submit)

◆ Advantages:

- ◆ Grid nodes choice is handled by gqsched taking into account replica catalog information, multiple nodes can be used
- ◆ Relatively simple
- ◆ File transfer is handled by gqsched
- ◆ One of the first advanced data schedulers!!!

◆ Disadvantages:

- ◆ Cost, feasibility, and complex requirements not yet considered

Based on our “ideal method” and our most important requirement, data access, we have developed a tool for distributing jobs to the grid. A job script together with a list of grid resources or gatekeepers is simply submitted using our tool. Security sign-on, file transfer, and the choice of grid nodes is handled by the tool. This also makes the splitting of jobs and access of multiple resources relatively simple. This may be the first data schedulers in existence as choice of grid nodes takes into account replica catalog information and data location.

The primary disadvantage is that this does not handle any complex requirements that a job might have, such as memory or software.

How it's supposed to work!



◆ A User Example (cont.) - GQSched script...

```
#!/bin/csh -f
#:Param FILE GridFile lfn:/users/winton/test.belle/*.mdst
#:Param EVTSKIP Numeric 0 to 9000 step 1000

#:StageIn recon.conf ; event.conf
#:StageIn particleTest.conf particle.conf
#:StageIn libanalyser.so ; user_ana.so ...

echo "Processing Job $JOBID on $FILE eventskip $EVTSKIP host ``hostname`
setenv FPDA_IO_PACKAGE fpdagrid.so
basfexec -v b20020424_1007 << EOF
path create main
module register user_ana
path add_module main user_ana
initialize
histogram define somehisto.hbook
process_event $FILE 1000 $EVTSKIP
terminate
EOF
echo Finished JobID $JOBID .

#:StageOut somehisto.hbook myoutput.${JOBID}.hbook
```

Here is an example of a script that can be submitted to the GQSched tool. It is just an ordinary Unix shell script with a few differences. Well formatted comments or directives are used to implement most features.

“Param” directives enable parameter sweeping for multiple jobs, the result of the parameters stored in environment variables. The example shows a “FILE” parameter which will iterate over all “*.mdst” files in the “/users/winton/test.belle” directory in the replica catalog. The \$FILE environment variable will be evaluated to the most appropriate physical file name (generally a URL) before execution on each host. The “EVTSKIP” parameter will numerically iterate from 0 to 9000 in steps of 1000. One job will be submitted for each FILE and EVTSKIP value and combination.

“StageIn” directives enable files to be transferred from your local host to the machines on which your jobs will execute. “StageOut” directives enable files to be transferred back to your local host and is generally used to pass back results.

An additional environment variable \$JOBID can be used to uniquely identify each job.

Overview



- ◆ Grids
- ◆ Data Grids
- ◆ Data Grids in practice
- ◆ Overview of products
- ◆ How it works
- ◆ HEP Projects <<<<
- ◆ ATLAS Data Challenges <<<<
- ◆ University of Melbourne
- ◆ Belle Experiment

HEP Projects



- ◆ What is High Energy Physics? (HEP)
 - Study of the fundamental constituents of matter and forces.
 - High Energy Physics - using H.E. enables the probing of smaller distances/structures and study in early-universe like environment.
 - Particle Physics - quanta of matter/forces and their properties

- ◆ How is this done?
 - Theories and Models to describe matter, forces, properties, actions, and interactions (Standard Model, CP violation)
 - Construct experiments/detectors and accelerators to investigate matter interactions and behaviour under high energy conditions.

- ◆ Day-to-day Experimental HEP



- The Construction - building an “Experiment” or “Detector” after proposal, investigation, R&D (typically many years)
- The Measurement - collection of data from many sensors within the Detector (many years)
- The Analysis - reconstruction of sensor data to summary data, correlation of sensor information; comparison with expected/simulated results; comparison with theory (many years!)

High Energy Physics (HEP) is the study of the fundamental constituents of matter and the forces between these constituents. It is called High Energy Physics as using high energies enables us to probe smaller distances and structures within matter, and also allows us to study matter as it was in the early universe, the history of matter. It is also called Particle Physics as we deal with quanta of matter and forces and the properties associated with these.

The study of HEP is broken into two main disciplines, theoretical and experimental. Theoretical HEP propose theories and models to describe matter, forces, their properties, actions, and interactions. Experimental HEP construct experiments or detectors and accelerators to investigate matter interactions and behaviour under high energy conditions.

Experimental HEP can be roughly broken into 3 separate activities. The boundaries of these activities, in time and responsibility, are often indistinct. The activities are the construction of detectors which typically takes many years, the measurement or collection of data, and the analysis of this data. We will focus on the using data grids for the analysis of data within HEP.

HEP Projects



- ◆ Day-to-day Experimental HEP - The Analysis
- ◆ Raw Data
 - individual “events”
- ◆ Simulated/Monte-Carlo Data
 - interaction/event generation
 - detailed detector simulation
- ◆ Reconstruction
 - correlation of sensor/subdetector information
- ◆ Reconstruction (as for raw data)...
- ◆ Data Summary for analysis [Belle 10TB]
- ◆ Skimming (subset) via loose “cuts” (parameter restriction) [Belle 100GB]
- ◆ Analysis to generate plots, histograms, Ntuples [Belle 100MB]
 - kinematic factors
 - topological factors
 - particle identification factors (based on assumptions)
 - track vertex fitting (based on assumptions)
 - particle/decay chain reconstruction (based on assumptions)
- ◆ Statistical Analysis - further cuts, fitting signal/bkg or theoretical prediction
 - signal = proportion correct assumptions; bkg = background/contamination
- ◆ Iterate above until analysis correct!
- ◆ Use for systematic error analysis.
- ◆ If vast differences/problems recheck simulation and analysis
- ◆ Systematic Error Analysis

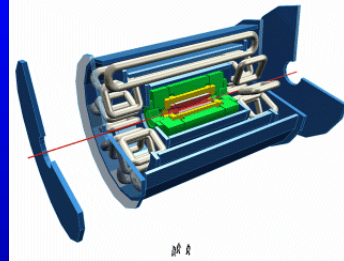
A typical analysis is split into two streams: data and simulation. Raw data is recorded from various sensors within a detector and stored as separate measurements or “events”. Simulated or Monte-Carlo data involves the generation of events and then detailed detector simulation. From this point on the analysis streams are very similar. The data is reconstructed which involves the correlation of sensor information. Data summaries are generated for ease of analysis. As an example, within the Belle experiment 10 TB of data summary information exists. These are “skimmed” to produce subsets of the data of most interest to each physicist’s analysis. This is done by applying, what we call, loose “cuts” which are restrictions on parameter space. These are around 100 GB in size for Belle users. These are then analysed to generate plots, histograms, and N-tuples. These can then be used for statistical analysis by applying further cuts and fitting signal, backgrounds, or theoretical distributions. For simulated data this process is repeated until the physicist feels their analysis is correct. The simulated data can then be used for systematic error analysis. The same analysis process is performed on data to obtain a result, provided there are no large differences between data and simulation.

HEP Projects



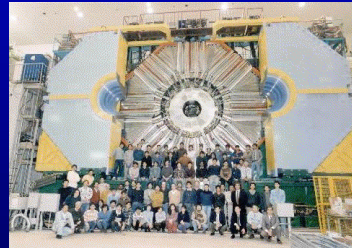
◆ The ATLAS Experiment

- Large Hadron Collider (LHC) at CERN, Geneva
- Search for Higgs particle which may lead to understanding the origins of mass.
- Collaboration 2000 people, 150 institutes, 34 countries
- 3.5 PB data per year
- operational in 2007



◆ The Belle Experiment

- KEK B-Factory, Japan
- Investigating fundamental violation of symmetry in nature (Charge Parity) which may help to explain the universal matter - antimatter imbalance.
- Collaboration 400 people, 50 institutes
- 10 TB data currently



The following two high energy physics projects are typical examples. The University of Melbourne Experimental Particle Physics group are participants in these collaborations.

The ATLAS Experiment will be built at the Large Hadron Collider facility at CERN in Geneva. The purpose of this experiment is to search for the Higgs particle which is of importance for understanding the nature of mass itself. It is a collaboration of 2000 people from 150 institutes across 34 countries. They expect to be producing 3.5 PB of data per year. The detector will be operational in 2007 and is currently under construction and development.

The Belle Experiment is situated at the KEK B-factory in Japan. It is investigating one of the fundamental violations of symmetry in nature, Charge Parity violation (CP violation). This will help use gain a better understanding of the matter - antimatter imbalance that is apparent in the universe. There are around 400 people in this collaboration from 50 institutions world wide. They currently have a data set of around 10 TB.

HEP Projects



- ◆ HENP groups (High-Energy and Nuclear Physics)
 - Relativistic Heavy Ion Collider (RHIC) at Brookhaven National Laboratory (BNL)
 - experiments: PHOBOS, BRAHMS, PHENIX, STAR
 - HENP Data Access Grand Challenge
 - » 50 TB data, 100's of simultaneous users, many institutes
 - » Mock Data Challenges arranged to test the systems
 - » RHIC has been running since mid 2000
- ◆ LHC computing grid (LCG)
 - Large Hadron Collider (LHC) at European Organization for Nuclear Research (CERN)
 - experiments: ATLAS, LHCb, ALICE, CMS
 - design and testing phase
 - each experiment will perform it's own "Data Challenges"

I'll mention a couple of HEP grid projects that we should be familiar with.

The first was lead by the High Energy and Nuclear Physics groups associated with the RHIC facility in the US. Several experiments were involved in what was called the HENP Data Access Grand Challenge which was to ensure access to 50TB of data for 100s of simultaneous users across many institutes. They created "Mock Data Challenges" to test their systems for readiness. The challenges were completed and successful and RHIC has been up and running since mid 2000.

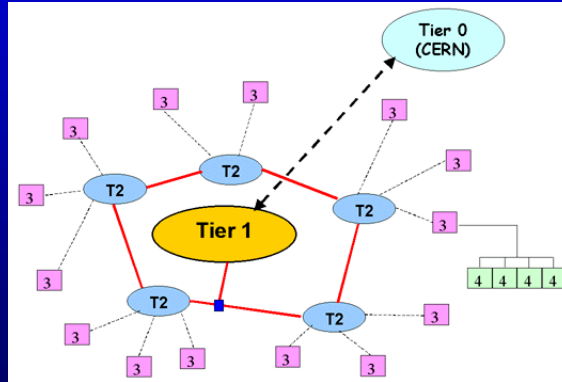
While the network of computing resources were not called a data grid, there are similarities. Many of the people who worked on this are now working on the next project, the LHC grid, and are bringing their experiences from RHIC to this.

The LHC grid will be constructed to service the experiments on the Large Hadron Collider facility in CERN and their collaborations. These are ATLAS, LHCb, ALICE, CMS. This grid is currently in the design and testing phase and each experiment will be performing their own "Data Challenges" to test the reediness of the grid systems for physics.

I will talk a little about the ATLAS data challenges further on, as we are participating in these.



- ◆ LHC Computing Grid (LCG) Infrastructure
 - The MONARCH Model (CERN computing division)
 - Hierarchy of Nodes (collections/clusters of computers)
 - Tiers 0, 1, 2, 3, 4, 5?



The infrastructure of the LHC grid will follow the “MONARCH Model”. The MONARCH group is a part of the CERN computing division and was set up to simulate and investigate the grid. This is the model they have recommended.

It consists of a hierarchy of Grid nodes or resources. Each node can exist on a number of Tiers or levels. The tier denotes the separation from the central CERN node (Tier 0). While this diagram shows just a single Tier 1 node, there will be quite a few of these, some situated within CERN itself.



◆ LHC Grid Infrastructure – The MONARCH Model (cont.)

		ATLAS expects	
Tier 0	CERN (1 node) connected to experiments	raw data first pass recon.	
Tier 1	1 per Experiment + Regional Centres	some raw data offline data additional recon. & sim.	>10 PB storage >1000 kSI95 ~10 nodes
Tier 2	Cities/Labs? Accessible to all.	partial offline data recon. & sim. support	400-1000 TB storage 300-500 kSI95 12-25 nodes
Tier 3	Institute Level Not accessible to all.	partial offline data private data	
Tier 4	Desktop / Personal	private data	
Tier 5	Mobile / Laptops / Transient	private data	

So, Tier 0 will be one node at CERN and will be connected to the experiments and store all raw data. This tier will do all of the first pass reconstruction of raw data to offline data. (Detector information and hits for each event are reconstructed into particle tracks which are more palatable for physics analysis.)

There will be 1 Tier 1 per LHC experiment situated at CERN plus regional centres situated in other countries. These may contain a little raw data but mostly all of the offline data. They will be used for additional reconstruction and simulation.

There will be many Tier 2 nodes situated at various laboratories and cities around the world. These will be accessible to all users within the experiments and will contain part of the offline data for reconstruction and simulation support.

Tier 3 nodes will be at institution levels and will not be accessible to the whole grid. They will contain some offline data and users private data.

Tier 4 nodes consist of personal desktop machines that are permanently connected to the grid. They will generally be used as portals to the grid and for the usual stuff.

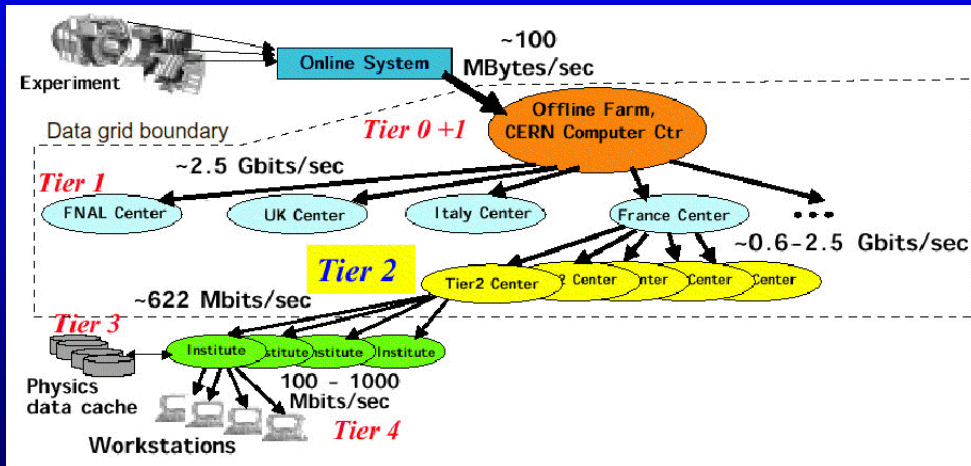
Tier 5 are the transient machines. And I'm not sure why they bothered going to this level of detail.

ATLAS expect to have about 10 PetaBytes of storage in total over all of tier 1 in about 10 nodes. They expect to have between 400 and 1000 TeraBytes of storage in Tier 2 amongst 12 to 25 nodes.

HEP Projects



- ◆ CMS expected grid hierarchy
 - CMS is a symmetric p-p detector at LHC looking for the Higgs
 - network connection between Tiers is very important



This is what CMS expect the grid to look like, from their point of view. One of the most important specifications within the grid hierarchy structure is the network connections between tiers. We need to be able to transfer the data to the correct level within the tiers, and we also need to be able to access this data from lower levels within the tiers.



- ◆ ATLAS expected requirements
 - 10^9 events per year
 - 3.5 PB data per year
 - » ~1.5 PB simulation data
 - » ~1 PB raw data
 - » ~200 TB ESD (event summary data)
 - » ~20 TB analysis data
 - » ~2 TB event tag data
 - 1000-1500 software users
 - ~150 simultaneous jobs
 - ~20 separate analyses
 - ~20 PB tape, ~2600 TB disk, ~2000 kSI95 in CPU
(1 SI95 = 1 SPECint95 \approx 40MIPS \approx P4 @ 20MHz)
(2000 kSI95 \approx 20,000 P4 2GHz)

This will give you an idea of the resources that might be involved in this infrastructure. ATLAS expects to take 10^9 events per year which will equate to 3.5 PB of data including simulation and analysis data. Over the period of the experiment they expect they will require 20PB of tape storage, 2600TB of disk storage and 2000kSI95 in CPU resources.

ATLAS Data Challenges



- ◆ Aims
 - Test readiness of infrastructure
 - Is the grid ready for physics?
 - Performance of computing hardware, network
 - Performance (efficiency and speed) of the software
 - Monitoring
 - Scalability
 - Test different structures: hierarchy vs uniform grid
- ◆ Specified 3 challenges (DC0, DC1, DC2)

So that we don't spend a large amount of money on computer resources and then discover there are fundamental problems, the experiments are proposing data challenges. These will test the infrastructure, network, hardware, the estimates that were made, and push the grid software forwards.

This also sums up the aims of the ATLAS data challenges.

They have specified 3 different data challenges which will build on each other over time. These called DC0, DC1, and DC2.

ATLAS Data Challenges



- ◆ Data Challenge 0 (DC0) Completed 8-Mar
 - Sample of 10^5 events in 1 month (trivial for the hardware)
 - “Continuity test” of the software chain
 - Including writing to / reading from persistent store
- ◆ Data Challenge 1 (DC1) Now!
 - Phase 0 April (preparation for phase 1)
 - Phase 1 May until 15-Jul (generate events for analysis)
 - » samples of up to 10^7 events in 10-20 days (20-30 TB)
 - » Physics goals: Re-check technical plots, find signal in event sample
 - Phase 2 Nov? until 2003? (software test, Geant4, Databases)
 - » Infrastructure goals: several hundred PCs world wide, test basic grid functionality (IO bottlenecks etc.)
 - » Physics goals: Pile-up simulation for high level trigger studies
- ◆ Data Challenge 2 (DC2) Starts in 2003
 - Sample of 10^8 events in 3 months
 - complexity ~50% of 2006-2007 LHC grid infrastructure
 - Infrastructure and Hardware: want to stress-test the model

Data Challenge 0 was completed in March and was designed to process a sample of 10^5 events in 1 month, which is trivial for existing hardware. It was mainly a test of the existing software and how it works together. It also included access to persistent storage across a grid. (Mass storage systems and remote disks.)

Data Challenge 1 which is underway now, is split into 3 phases. Phase 0 was preparation. Phase 1 is the generation of 10^7 events in 10-20 days. ATLAS will also use this to test some physics of the detector and analysis techniques. We are currently participating in these data challenges together with the Melbourne Advanced Research Computing Centre. They have allocated 16 CPUs for this and a further 16 CPUs for an identical grid node for other experimental HEP use.

Phase 2 will test different software solutions and the infrastructure of the grid. (Stress testing.)

Data Challenge 2 which is to occur in 2003, will construct and process a sample of 10^8 events in 3 months. This will also test the system at about 50% the complexity of the 2006-2007 LHC grid system from ATLAS perspective. This will mainly test the infrastructure and hardware that will be used.

Overview



- ◆ Grids
- ◆ Data Grids
- ◆ Data Grids in practice
- ◆ Overview of products
- ◆ How it works
- ◆ HEP Projects
- ◆ ATLAS Data Challenges
- ◆ University of Melbourne <<<<
- ◆ Belle Experiment <<<<



- ◆ Collaborative Grid Activities
 - Melbourne Advanced Research Computing Centre (MARCC; UofM HPC facility)
 - Dirk van der Knijff, Robert Sturrock, Paul Edwards, Bryan Hellyer
 - » Data Grid infrastructure for HEP
 - » Currently taking part in the ATLAS DC
 - Computer Science (UofM)
 - Leon Sterling, Muthukkaruppan Annamalai
 - » Ontological framework for experimental HEP analysis
 - » High-level descriptions of analysis for use by analysis code/agents
 - Computer Science, GridBus (UofM)
 - Rajkumar Buyya, Shoaib Ali Burq, Srikumar Venugopal, Steve Melnikoff
 - » Resource Brokering and Economic Scheduling for the Grid and HEP
 - » Best/cheapest use of available Grid resources.

At the University of Melbourne we are participating in a number of collaborative Grid activities.

We are working with the MARCC group (the Melbourne Advanced Research Computing Centre) headed by Dirk van der Knijff to help build a data grid infrastructure for use within HEP. We are currently taking part in the ATLAS data challenges mentioned previously.

We are working with members of Computer Science department headed by Leon Sterling in developing an Ontological framework for experimental HEP analysis. This is a framework for high-level descriptions for use by analysis code or agents.

We are working with another group within the Computer Science department, called GridBus, headed by Raj Buyya, in developing tools for resource brokering and economic job scheduling for use within HEP. This effectively means the ensuring the best and cheapest use of available grid resources.



- ◆ Collaborative Grid Activities (cont.)
 - Computer Science (RMIT)
 - Lin Padgham, Wei Liu, Antony Iorio
 - » Agent based technology for experimental HEP Analysis
 - » Intelligent services able to dissect, suggest, and perform analyses.
 - GrangeNet & APAC (ANU, Canberra)
 - Jon Smillie, Stuart Hungerford, Markus Buchhorn
 - » Data storage with Mass Data Store facility (petabyte storage)
 - » GrangeNet - Australian gigabit research network
 - » Possible use of future Grid computing resources ?
 - Victorian Partnership for Advanced Computing (VPAC)
 - Bill Appelbe, Sudarshan Ramachandran
 - » Possible use of future Grid computing resources ?
 - » TRASC - simple platform-independent access to the Grid

Outside of the University, we are working with members of the Computer Science department of RMIT, headed by Lin Padgham, on developing agent based technologies for HEP analysis. These will be intelligent services able to dissect, suggest, and perform analysis based on high level descriptions of the analysis.

GrangeNet and APAC have provided us with access to their Mass Data Store facility in Canberra. This is a petabyte storage facility and will be used to store Belle analysis data. We are hoping to have access to the GrangeNet gigabit research network to enable faster, cheaper access to this data. There may also be some future grid computing resources available for our use.

The Victorian Partnership for Advanced Computing (VPAC) may also be implementing grid computing resource that could be of use. They have also developed a utility, TRASC, which we are investigating.



- ◆ Local Grid Activities (Experimental Particle Physics group)
 - VPAC (Victorian Partnership for Advanced Computing) funding to build expertise and resources in High Performance Computing and Data Grid technologies within HEP
 - Existing HEP analysis within Grid environment (Belle experiment)
 - » Belle is situated at KEK B factory (Japan) researching CP violation in standard model (400 users; data set ~10 TB)
 - » Implementation of Grid architecture to enable collaborative access to resources/data for an existing physics application
 - » Important contribution to the HEP and Grid communities by providing a “real-life” application
 - Grid Software expertise and development
 - » Expertise in Grid resource software and deployment to take advantage of collaborative computing is important for future HEP research in Australia (most facilities are overseas)
 - » Software development for the wider HEP/Grid communities... physics driven applications for job control and data manipulation

Within the Experimental Particle Physics group we are also undertaking a number of local Grid activities. We have obtained funding from our generous benefactors, the Victorian Partnership for Advanced Computing, to build expertise and resources in High Performance Computing and Data Grid technologies within HEP.

In particular we've identified 2 major activities. The first is to investigate the advantages of using the Grid for HEP analysis within the Belle experiment. The Belle experiment is situated at the KEK B factory in Japan and is to research CP violation in the standard model. We are implementing a grid architecture to enable collaborative access to resources and data for an existing physics application. This is an important contribution as it will provide a real-life test-bed for the use of the Grid within HEP.

The second major activity is to gain software expertise and to develop software for the Grid and HEP. Expertise in Grid software will help us take advantage of collaborative computing and is important for the future of HEP research in Australia as most facilities are situated overseas. Contributing to software development enables us to develop user driven applications designed for HEP. These activities overlap to a large extent.

Belle Experiment



- ◆ Belle Analysis Data Grid (BADG)
 - enable Belle analysis within Australia to run in Grid environment
 - if useful locally » adopted by Belle » wider community
- ◆ Goals for BADG
 - ☑ – construction of a Grid Node at Melbourne
 - » Certificate Authority to approve security
 - » Globus toolkit...
 - ◆ Globus Gateway - connected to local queue (PBS queue)
 - ◆ GSIFTP - data resource providing access to local storage
 - ◆ Replica Catalog - LDAP for virtual data directory
 - ☑ – initial test of Belle code with grid node & local queue
 - ☑ – modification of Belle code to access the data on the grid (physical file names as stored in Replica Catalog)
 - ☑ – test of Belle code with grid node & queue & grid data access
 - ☑ – connect 2 or more grid nodes (Melbourne EPP, Sydney)
 - ☑ – test of Belle code running over grid of 2+ nodes
 - ☑ – implement or build a Resource Broker and/or Scheduler
 - active use by physicists ; test tools and stability

Focusing on these activities, I'll look at enabling our Belle analysis within a Grid environment within Australia. We hope that this will provide a more effective use of the computing resources available to Belle in Australia. If we are able to prove that this is useful locally, this may be adopted by the Belle collaboration. On a wider level, actually using the grid infrastructure for real physics today, can be an important test for others looking at using it in the future.

So what needs to be done:

Firstly we need to construct a grid node; then we need to do initial tests of the Belle software without modification to test the grid resources; then modify the Belle analysis code to access data on the grid. This needs to be tested together. To date, we have completed these initial tasks and are looking at the next steps.

We then need to connect multiple grid resources and then test the Belle code over these. And finally implement or build a Resource Broker or Scheduler to help better utilise our grid resources. Once this is done we hope to have our grid in active use by physicists and to continue to test the tools and stability.

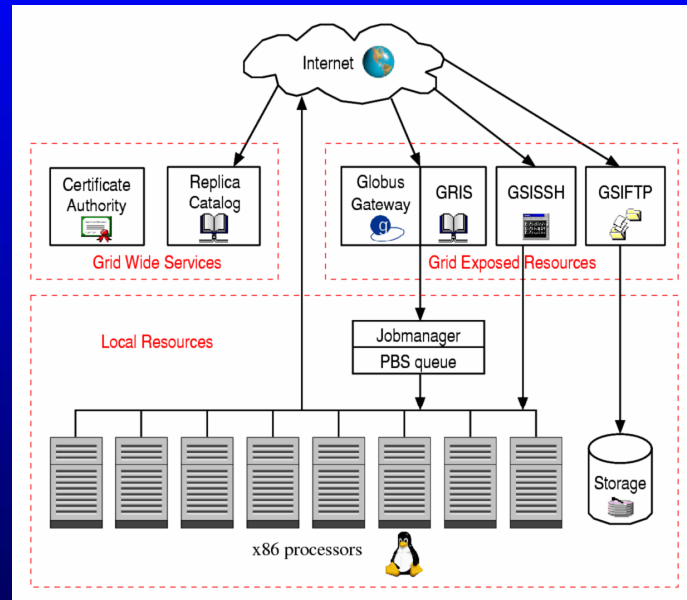
To-date all but the last 2 goals have been completed. The implementation of a resource broker or scheduler is partially complete but requires more development and the investigation of other products.

Of course, if the decision is made to extend the grid throughout the international Belle collaboration then there is much more work to be done!

Belle Experiment



◆ Melbourne Experimental Particle Physics Grid Node



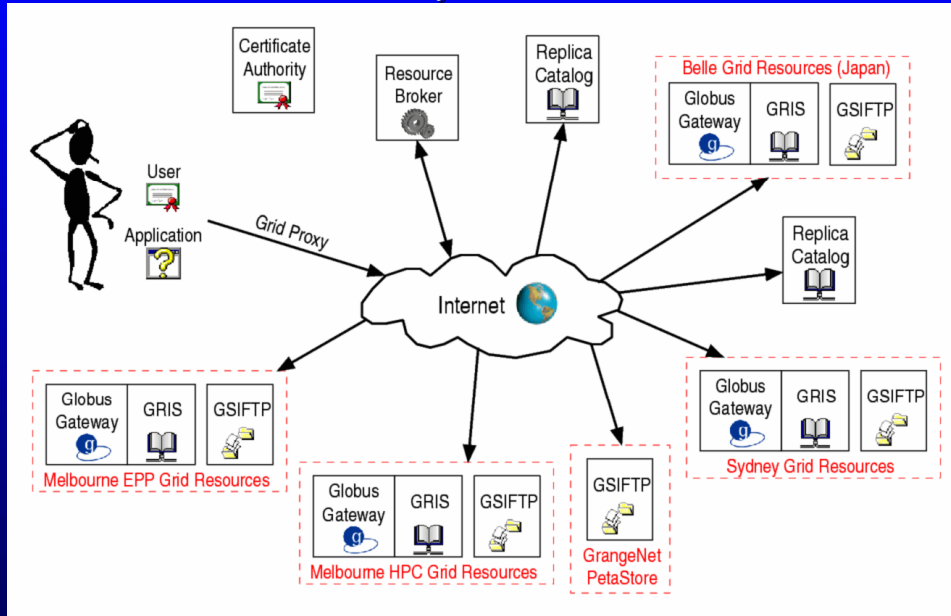
Here is an overview of the our Grid node at Melbourne University...

Our existing resources consist of a number of x86 processor machines running Debian Linux accessible via a PBS queue system, and around 200Gb of disk storage. We have exposed these as Grid Resources by installing a Globus Gateway and Grid Resource Information System, and a GSIFTP server for data access. There is also a GSISSH entry point for interactive access. The Globus Gateway accesses the PBS queue system via a customised Jobmanager script. We've also implemented some Grid wide services such as a Certificate Authority to approve user and host certificates, and a Replica Catalog for the filing and tracking of data. The idea is that our local resources are accessible to anyone on the internet as Grid Resources. Users and applications on our local resources can access our Grid Resources and Other Grid Resources via the internet.

Belle Experiment



◆ The future Belle Analysis Data Grid



In future we need to incorporate other Grid resource to construct a HEP Belle analysis Grid. Through our collaboration with Sydney University and the Belle experiment we have grid enabled their resources for our mutual use. In addition GrangeNet have provided us with access to their petabyte storage facility located in Canberra which will be accessed over the new GrangeNet gigabit network. In the future it is hoped that our collaboration with Melbourne Uni's High Performance Computing facility (the MARCC centre) will provide us with access to some of their computing and storage resource. There may also be several cloned grid wide resources such as replica catalogs at various locations.

A Grid user is identified by a certificate signed by the certificate authority. If you recall our user examples of running Grid jobs, a user signs on to the grid by creating a grid-proxy. The user will submit jobs using the grid proxy via the internet to grid resources. When the job applications is running on these resources it inherits the grid proxy of the user, allowing it to further utilise grid resources via the internet.

Eventually we may incorporate a centralised Resource Broker into our Grid to manage the matching of jobs with resource, availability of resources, and to help schedulers use the grid more efficiently for many users.

Belle Experiment



- ◆ Belle analysis test case...
 - Rohan Dowd, Ph.D. student at Uni.of Melb.
 - » Analysis of charmless B meson decays to 2 vector mesons, used to determine 2 angles of the CKM unitarity triangle (ϕ_2 , ϕ_3). These decays have not been measured previously.
- ◆ Rohan's analysis code (test of 10 files ; 2 GB total)
 - Data files processed serially 95 mins
 - Data files processed over Globus 35 mins
- ◆ Data access (2 secure protocols GASS/GSIFTP ; 100 Mbit network)
 - NFS access for comparison 8.5 MB/s
 - GASS access 4.8 MB/s
 - GSIFTP access 9.1 MB/s
- ◆ Belle analysis using Grid data access
 - NFS access for comparison 0.34 MB/s
 - GSIFTP data streaming 0.36 MB/s

We decided to use an analysis test case which is the analysis of Rohan Dowd, a Ph.D. student at the University of Melbourne. His analysis is investigating charmless B meson decays to 2 vector mesons. This is used to determine 2 angles of the CKM (Cabibbo-Kobayashi-Maskawa) unitarity triangle. These types of decays have not been measured previously.

We've performed a few preliminary tests of the analysis and the Grid. The first test was processing of 10 sample files, a total of 2GB. When processed serially all files took 95 minutes. When processed using our facilities as a Grid Resource the files took 35 minutes to process. This is not surprising as the files are processed in parallel and the time is limited by the longest file.

Next we tested the 2 available data access protocols GASS and GSIFTP across our 100 Mbit network. For comparison NFS access to files is around 8.5 MB/s. File access via GASS is 4.8 MB/s and GSIFTP is 9.1 MB/s. Based on this initial test we've decided to use GSIFTP as it appears to have less overhead.

Combining the two tests, we ran the Belle analysis using Grid data access via GSIFTP and found very little difference in performance compared to running of NFS. So we can tell, this particular analysis is CPU limited. So the Grid does not degrade the performance of our test analysis.

An important note: For the application we are using we do not transfer the entire code as this is around 1GB. For grid processing we must ensure the application is installed on any node we use and transfer users analysis as shared object libraries which can be plugged into the application.

Belle Experiment



- ◆ Belle analysis production case...
 - Jeremy Dalseno, Honours student at Uni.of Melb.
 - » Analysis of simulated B decays to $D^*D^*K_S$ used to resolve 1 angle of the CKM unitarity triangle (ϕ_1).
- ◆ Jeremy's analysis code (before optimisation)
(1 file ; 400 MB total ; split into 20 pieces)
 - File processed sequentially ~ 48 hours
 - File processed split up over Globus ~ 5 hours
- ◆ What we've learnt...
 - splitting of "large" files or grouping of "small" files
 - greater job control and monitoring is required
 - disk access and NFS might become a problem (many files accessed from one central server; local resource issue)
 - need ways to specify complex requirements so we can better manage memory, disk space etc.
 - need to track Belle software versions as analysis code is specific (many versions; too large to transport with job; restricts sites)

Our first production case was to use the analysis of Jeremy Dalseno, an honours student at the University of Melbourne. His analysis is investigating simulated B meson decays to $D^*D^*K_S$. This is used to resolve 1 angle of the CKM (Cabibbo-Kobayashi-Maskawa) unitarity triangle.

The grid analysis required the processing of 1 file, only 400 MB in size, however this took a couple of days to process sequentially. When split into 20 sections and processed via the grid this took around 5 hours in total which is as expected. However, Jeremy's analysis proved to be a major stress test for our system due to the memory size of the jobs and disk access.

We learned a number of things from this exercise. First, that the splitting of "large" files and conversely the grouping "small" files will become a factor in the way we utilise the grid. (Small and large pertain to processing time more than actual file size.) Greater job control and monitoring is required than is provided by the currently available tools. Disk access and NFS might become a problem as within this analysis NFS was use for all input and output. We need to find better ways to manage our local resources including memory to protect against overuse. This might be solved by investigation into local job management software.

As a final note: For the application software we are using we do not transfer the entire code as it is gigabytes in size. So for grid processing we must ensure the application is installed on any node we use and transfer users analysis as shared object libraries which can be plugged into the application. Many versions of the Belle software exists and we may need to track these.

Belle Experiment



◆ Problems Encountered

- Grid middleware Globus needs a lot of work
 - » many bugs (each grid admin has their own favourite list)
 - » documentation is poor and lacks troubleshooting
 - » interfaces, protocols, structures are developing rapidly & unstable
 - » APIs are heavy weight
 - » various & large problems for non-Linux platforms (Solaris, Tru64...)
- Certificate Authorities and Security
 - » what structure to use? Globus CA / Private CA / National CA
 - » signing policies allow insecurity and ambiguous usage
 - » future “virtual organisation” services may solve some problems
- How to best utilise our local resources
 - » Limited number of supported job management systems
 - » OpenPBS with Maui scheduler and SSH communication
- Mass Storage Systems
 - » protocol timeout during tape staging
 - » how to manage file staging efficiently (many files + distributed jobs)
- Firewall issues
 - » file staging and stdin/stdout requires access to submission host

To go over some of the problems that we’ve encountered...

The Globus toolkit, the foundation middleware, still requires a lot of work. There are many bugs; the documentation could be improved particularly where troubleshooting is concerned; the code is developing rapidly which generally means it’s unstable; the APIs are heavy weight and can lead to problems; and the problems seem larger for platforms other than Linux.

Security and authorisation is still an issue within grid communities. This is mainly to do with policies regarding which Certificate Authorities. Much of this will be solved by the introduction of “virtual organisations” which is in the near future.

We have come across a number of issues about how to best utilise our local resources. There are a limited number of job management systems supported by Globus and our original system was not one. We’ve finally gone with the combination of OpenPBS and Maui but other systems might be more appropriate.

Access to Mass Storage Systems such as the GrangeNet petabyte storage facility has a few problems. This will probably continue over time as we learn more about our data access patterns.

And there are a few firewall problems that are yet to be resolved.

Summary



- ◆ The Grid will play an important role in collaborative research
 - the technology is a long way from finished
- ◆ ATLAS Data Challenge is underway at Uni. of Melb.
 - provide with exposure to other/future grid environments and packages
- ◆ Belle Analysis Data Grid within Australia
 - construction of a grid infrastructure is underway
 - successfully utilised grid resources in an analysis test case
 - further work is required for production analysis
- ◆ Development of generic data grid tools
 - tools for querying and maintenance of Replica Catalog
 - tools for scheduling of jobs over multiple nodes (GQSched)
 - further work for job control and maintenance
- ◆ Make the experience gained accessible to the HEP (and perhaps wider) community for the betterment of collaborative research
 - looking for others who might be interested in Grid computing

In summary, the Grid will play an important role in collaborative research and in particular, the future of HEP. However, the technology is a long way from finished.

The ATLAS Data Challenges are underway at the University of Melbourne. These challenges will increase our understanding of the future of HEP computing, and the results will aid in the development of Grid software and the LHC Grid infrastructure.

The construction of a grid infrastructure for the Belle experiment analysis is well underway. We have successfully used Grid Resources in an analysis test case but are yet to demonstrate the potential benefits of shared resources.

The development of a generic Grid tools is an ongoing task. In our opinion contributing to the wider community in this project and in other ways is the best way to gain experience with Grid computing. Our focus has been on tools for the querying and maintenance of data within replica catalogs and the scheduling of jobs over multiple nodes.

But, my overall goal is to make the experience we've gained accessible to the HEP and perhaps wider community, as we have a lot to gain from collaborative research. And finally, we are looking for others who might be interested in grid computing.

References



- ◆ University of Melbourne, Experimental Particle Physics group
 - <http://www.ph.unimelb.edu.au/epp/>
- ◆ Victorian Partnership for Advanced Computing (VPAC)
 - <http://www.vpac.org/>
- ◆ Melbourne Advanced Research Computing Centre (MARCC)
 - <http://www.hpc.unimelb.edu.au/>
- ◆ University of Melbourne, Department of Computer Science & Software Engineering
 - <http://www.cs.mu.oz.au/>
- ◆ The GridBus project (Grid Computing and Business)
 - <http://www.gridbus.org/>
- ◆ Monash University, School of Computer Science & Software Engineering
 - <http://www.csse.monash.edu.au/>
- ◆ BELLE Collaboration
 - <http://belle.kek.jp/>
- ◆ ATLAS data challenges
 - <http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/DC>
- ◆ LHC Computing Grid project
 - <http://lhcgird.web.cern.ch/>
- ◆ Globus project
 - <http://www.globus.org/>
- ◆ European Data Grid project (EDG)
 - <http://www.eu-datagrid.org/>
- ◆ Grid Data Mirroring Package (GDMP)
 - <http://project-gdmp.web.cern.ch/project-gdmp/>
- ◆ GriPhyN - Grid Physics Network
 - <http://www.griphyn.org/>
- ◆ Particle Physics Data Grid (PPDG)
 - <http://www.ppdg.net/>
- ◆ Nile - National Challenge Computing
 - <http://www.nile.cornell.edu/>
- ◆ Sun™ Grid Engine
 - <http://www.sun.com/software/gridware>