

CS11 San Diego Workshop, 20-21 March 2003

Workshop home page:

<http://www.ppdg.net/mtgs/20mar03-cs11/index.html>

Present: David Adams, Craig Tull, Tony Johnson, Doug Olson, Conrad Steenberg, Maarten Balintijn, Rob Gardner, Eric Aslakson, Harvey Newman, Wayne Shroeder, Torre Wenaus, Joseph Perl

Thursday, 20 March 2003

Doug's intro to Web Services and OGSA.

<http://www.ppdg.net/mtgs/20mar03-cs11/ogsa-summary-olson.pdf>

- Questions about why WSDL versus Corba. Web Services more for component design, not all parts of the overall system from same producer (though Corba can probably provide much of this same sort of discovery mechanism).
- Issues of XML speed. Every web service message means setting up and tearing down tcpip sockets. But can keep socket open longer. Might switch to some other protocol for parts of communication that need to be specialized/fast.
- Web Services addresses persistent services. In Grids, must also address transient services.
- Early days for this technology. Toolset to make it easy isn't there yet.
- Nice functional example OGSA database application from <http://www.ogsadai.org.uk/>
- Virtual Organization Registry Service Pattern
- OGSA needs Globus 3. Currently in alpha release.

Doug also showed a talk by ogsadai.org.uk from GGF7, Tokyo, titled "Data Access and Integration".

<http://www.ppdg.net/mtgs/20mar03-cs11/dai-summary-olson.pdf>

- Analyst goes to Registry get handle of service factory
- Analyst then goes to Factory to have it create service and return handle of service
- Analyst then asks Service to perform function, returning data either to analyst or to some other customer.
 - Joseph asked whether anything is specified about who might keep track of the console log of the service, to have it available for an admin to browse, during session or perhaps after a crash. Also questions about who might be responsible for killing the service if needed. Might be done by the service factory, might be done by some other service.

Reworked API Diagram that had been discussed at last (Caltech) workshop.

Purpose of this diagram is not to show exact architecture (there is no one architecture that represents all possible systems for interactive analysis on the grid) but to find a more or less complete list of what grid APIs/services are needed by such interactive analysis. A later step is then to take these APIs one at a time to search for an existing API/ extend an API or create a new API to meet these needs.

Final version from this workshop is shown at:

http://www.ppdg.net/mtgs/20mar03-cs11/APIsDiagram_rev3.pdf

http://www.ppdg.net/mtgs/20mar03-cs11/APIsDiagram_rev3.ppt

David: Dial

- sequence diagram (from CHEP03 DIAL presentation)
<http://www-conf.slac.stanford.edu/chep03/register/report/abstract.asp?aid=388>
- Where our API diagram has “Analysis Tool”, DIAL has a Client Scheduler and a Server Scheduler
- To get a good estimate, have to have essentially done the match making
- There’s a lot inside grid scheduler
- Dataset has an event list and a content list

Discussion of how to handle partial results when one of n nodes crashes. JAS model is to just fetch binned histogram data, not raw data, for partial results. If job crashes, this model naturally has one get fresh version of partial results for that data set.

Might want estimator to be able to work other way around: “supposing I only want to spend ten seconds, how much data can I analyze?”

Matchmaker: tries to match computing resources to data resources. Has some policy built into it.

Need to talk about what kind of architecture we can expect. For example, are farm nodes going to be allowed to talk to the client elsewhere.

Maarten: PROOF

- Sequence diagram (photo of whiteboard)
<http://www.ppdg.net/mtgs/20mar03-cs11/pix/pix2.JPG>
- Client creates master, creates slaves
- Master sends files to slave that contain selector
- Slave asks master for a new packet when it is ready
- Reports stats back to master when asking for new packet (can be used for progress monitoring)
- Could deal with latency issues between master and slave by having master queue up a series of work packets on the slave
- Using the grid for the create phases and file transfer
- Otherwise plain tcp, simple messages, keeps it simple, don’t have to link to anything, but does have firewall issues (could be solved by passing that through a higher level protocol)
- Would be nice to be able to move the data around at a higher level than files (such as file sets).

Scheduler issues have come up repeatedly in our discussions. How do interactive jobs co-exist with batch jobs on a grid scheduler? If the interactive analysis applications

implement their own schedulers, how can these work properly with the standard grid scheduler? Tradeoffs are something like networking service guarantees.

Does Resource Broker handle decisions about Network Resources as well as CPU?

Tony: JAS

- Sequence diagram
<http://www.ppdg.net/pipermail/ppdg-idat/2003/msg00044.html>
- This is a “conceptual” sequence diagram
 - For discussion purposes only
 - Corresponds to existing JAS/Grid prototype
 - Greatly simplified, most interactions shown are really composed of many sub-interactions
 - Not real method names
 - This is an interactive system, user can do things in many different ways, this is just (a very simple) example
- Drill down through dataset catalog hierarchy
- Get info about dataset (such as “which version of MC was used to generate this data”)
- Dataset is a logical file which may exist as a set of physical files, but might also be a query in a relational database or a set of objects in an object database
- Just require that the dataset catalog can be presented as a hierarchy and can give you an “event source”
- Have simpleminded data catalog and resource broker waiting to be replaced by something better provided by the grid
- Resource broker and data catalog communicate with each other

Tried to write a sequence diagram for a generic interactive analysis session on the grid. Found that we had a lot of differences about what work is done in a “scheduler” versus what is done elsewhere.

- User application asks
- Dataset catalog for list of
- Datasets
- And passes that list and some required
- Response characteristics (such as desired completion time) to a
- Scheduler (aka, abstract concrete planner)
- To look at it another way, the Scheduler gets a Job that consists of
 - Application
 - Dataset
 - Response characteristics which talks to
- Scheduler gives job to a
- Match maker that asks questions of
- Sites

Friday, 21 March 2003

Quickly reviewed Rob's talk, ATLAS Grid Component Library (difficult to discuss much of this since Rob was unable to attend this day due to other writing commitments)

<http://www.ppdg.net/mtgs/20mar03-cs11/rwg-gcl-v2.ppt>

Continued to work on brief descriptions of the full set of APIs.

Required defining terms:

- "Abstract Job" = dataset + algorithms (what the user sets up).
- "Concrete Job" = single invocation of abstract job at user level.
- "Subjob" = part of a concrete job sent to a specific executor.
- "Dataset" = a collection of data that can be processed event by event. Has to provide a means to gather the data for each event. Should have a unique identifier, ideally with option to produce a human readable identifier.

Final version from this workshop is shown at:

<http://www.ppdg.net/mtgs/20mar03-cs11/APIsBriefDefs.doc>

Tony: Straw man dataset catalog interface.

This is what we would like a grid dataset catalog to provide for us so that our interactive analysis tools could give the user a good experience of the dataset catalog. As agreed, we wrote this interface as java pseudo-code. We imply nothing about the actual implementation, other than that we try to come up with an interface that is capable of being implemented.

Workshop developed two versions of this Dataset Catalog Service Query Interface. Second version is a reworking of the first one but trying out the idea that perhaps the Dataset Catalog is itself just a specific kind of Metadata Catalog.

<http://www.ppdg.net/mtgs/20mar03-cs11/strawman.java>

<http://www.ppdg.net/mtgs/20mar03-cs11/strawman2.java>

Still don't have this interface to the level that we can all use. Is it useful to continue these discussions unless we can get this interface to the level that we can all use (and thus be able to interoperate at least in this limited way)? Should we try working on a different one of these interfaces for a while?

For Martin and Tony, solving the dataset catalog interface would solve a specific problem that currently exists in PROOF and JAS.

Proceeded to break Dataset into several subtypes which might need different catalog service interfaces.

- **Physicist/User Dataset:** "The CMS higgs candidate minidst events from 2010 with missing Energy > 20Gev"
 - Must be a snapshot of a query result (i.e. constant in time)
 - Provenance should include query and timestamp
 - Must include way to access events.

- **Logical Dataset:** "List of (app/exp specific) record (e.g. event) id's + description of content (e.g. tracks+jets) + maps to a specific set of bits that can be instantiated"
- **Logical File Dataset:** Logical dataset mapped to (≥ 1) logical files.
- **Hierarchical Dataset:** Logical dataset mapped to > 1 logical dataset.
- **SplitDataset:** A Logical dataset split into multiple pieces, the sum of which are equivalent to the original dataset. Ideally dataset has natural places where it can be split.

Datasets may result from queries such as "Give me the CMD higgs candidate events from 2010 with missing energy $> 20\text{Gev}$ ". Want to be able to store those queries, although the same query may give a different answer at a different time.

Looked at FreeHEP SequentialRecordSource interfaces (developed by Tony, Simon Patton, Mark Donzselmann, Joseph), trying to adapt the JAS2 datasource model to the requirements of JAS+Wired+LCD+IceCube.

See package org.freehep.record.source at:

<http://java.freehep.org/lib/freehep/api/index.html>

- Layered model with base layers not specific to problem domain but increasingly problem-domain-specific layers.
- Any or all of our Types of Datasets might implement one of these SequentialRecordSource interfaces.

In reporting back to other grid services, it is important that we remember to point out both our long term needs and our shorter term needs. Discuss what is the minimum that we need right now, but also keep an eye on what we will be asking for in the longer term.

Discussion of whether Dataset catalog is one particular kind of metadata catalog and that therefore there might be no reason to have separate Dataset Catalog Service API and Metadata Catalog Service API. Counter-argument is that having both APIs doesn't preclude one choosing to implement the Dataset Catalog Service API through one's Metadata Catalog Service. Metadata catalog could be useful for other kinds of queries such as about jobs rather than about datasets.

David: Can we find a common scheduler.

Give scheduler a job =

1. LogicalDataset (probably also LogicalFileDataset and HierarchicalDataset)
2. Application (exe, task input files, ...)
3. Response time

Creates a concrete plan: Scheduler splits dataset onto sub-datasets and matches them with its sub-schedulers to define sub-jobs.

-

What we really want to get from the grid is the planning part of this work. There is interest in trying to share such a thing.

Where do we go from here:

- Think and iterate on the work we've done here, specifically the Dataset Catalog Service Query API.
- Try to translate our Java pseudo-code to WSDL.
- Continue to study our list of APIs.