

A Grid Service for Interactive Dataset Analysis

6th International Workshop on Java for Parallel and Distributed Computing

IPDPS Meeting April 26, 2004 – Santa Fe, NM



David A. Alexander, Balamurali Ananthan, & Brian Miller
Tech-X Corporation



Tony Johnson, Max Turri, & Victor Serbo
Stanford Linear Accelerator Center

The US Dept. of Energy (Grant DE-FG03-02ER83556) funds this work at Tech-X Corporation in collaboration with the Stanford Linear Accelerator Center



Specialized System for Parallel Data Analysis



- In high energy physics parallelism is straightforward because data is a collection of many files which are a set of independent events. Community is distributed
- Useful analysis is complex and researchers need feedback, thus interactivity is desirable
- Java Analysis Studio (JAS) is a combination IDE and graphical tool -- powerful single file analysis capability.
- Project described here is a Phase II Small Business Innovative Research (SBIR) grant from the Office of Science/HEP of DOE.



Project Goal and Driving Ideas

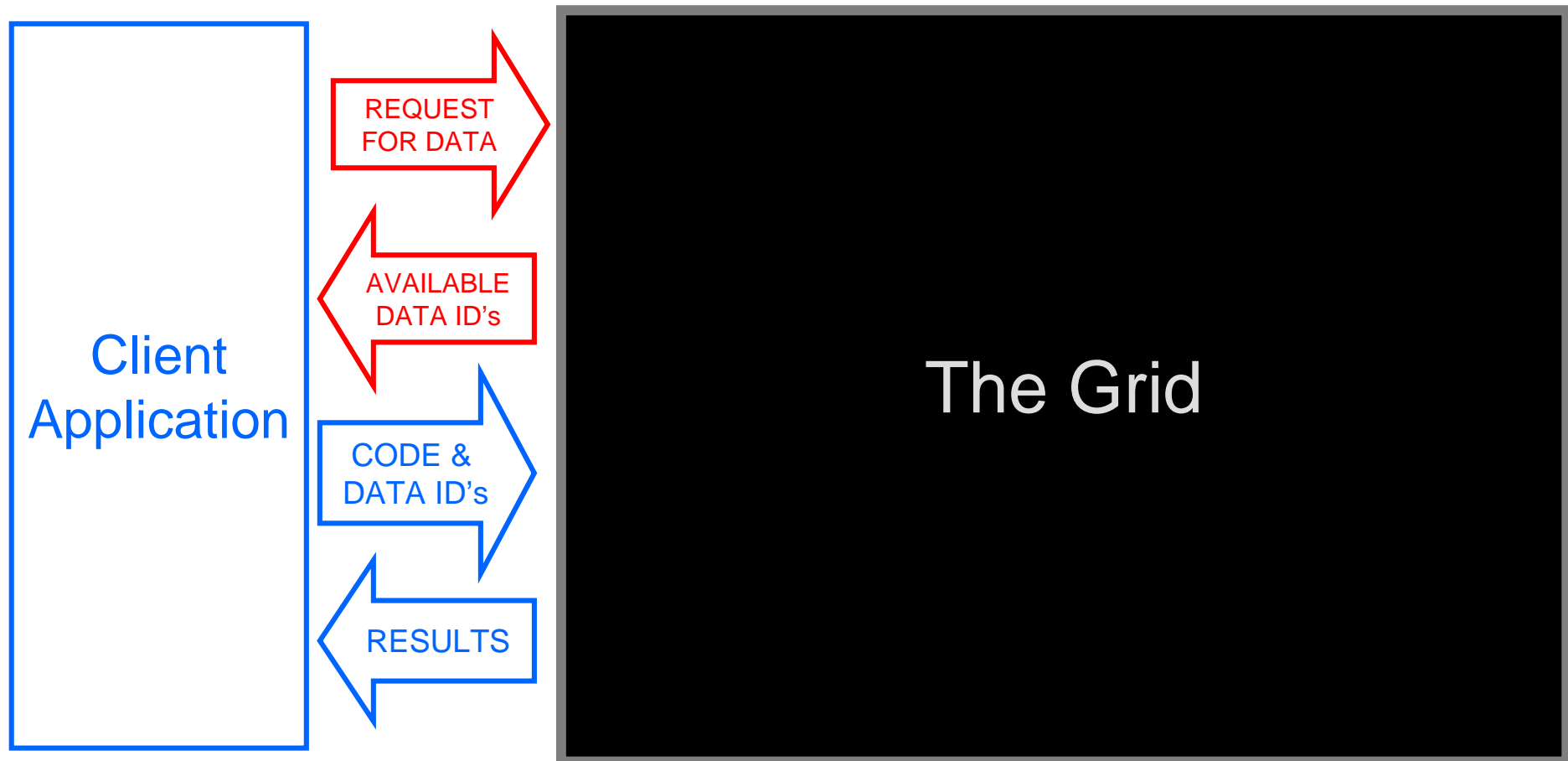


- Goal is to create a set of Grid services for clients like JAS to analyzing large data sets. Important ideas are:
 - **Interactive not batch processing**
 - Users write own analysis code to be run on many large virtual datasets while they develop the code.
 - Visualizations fill while analysis job is executing. Results appear in time scales of less than a second.
 - Can pause, alter, and restart analysis job at any time.
 - **Grid Service Infrastructure**
 - Access security provided (X509 Certificates)
 - Component structure easier to develop and flexible to connect
 - Provides interconnectivity between different implementations through well-defined interfaces and is independent of language (Client applications from different authors can use same service, visa-versa)
 - **Standardization of Common Interfaces into Grid Services**
 - Work through *Particle Physic Data Grid (PPDG)* interactive data analysis working group



Client Deals with Dataset ID and Analysis Code

Stanford
Linear
Accelerator
Center





This Talk is About Prototype and Planned Work



- Grid Prototype System working
 - Demonstrates concept with pre-Grid-services Globus Toolkit GT2
 - Uses JAS version 2, distributed model
 - Uses Java Remote Method Invocation (RMI) as the communication mechanism.
- GT3-based system designed and in progress
 - Current effort underway to define Grid service interfaces with PPDG & we have a reference implementation of a dataset catalog service



Prototype has Grid Job



The screenshot displays the Java Analysis Studio (JAS) application window. The main window has a menu bar (File, Edit, Job, Histogram, View, Window, Help) and a toolbar with various icons. The main content area shows a "Welcome to Java Analysis Studio" message with buttons for "Welcome To JAS", "JAS Tutorial", and "Creating a New Job".

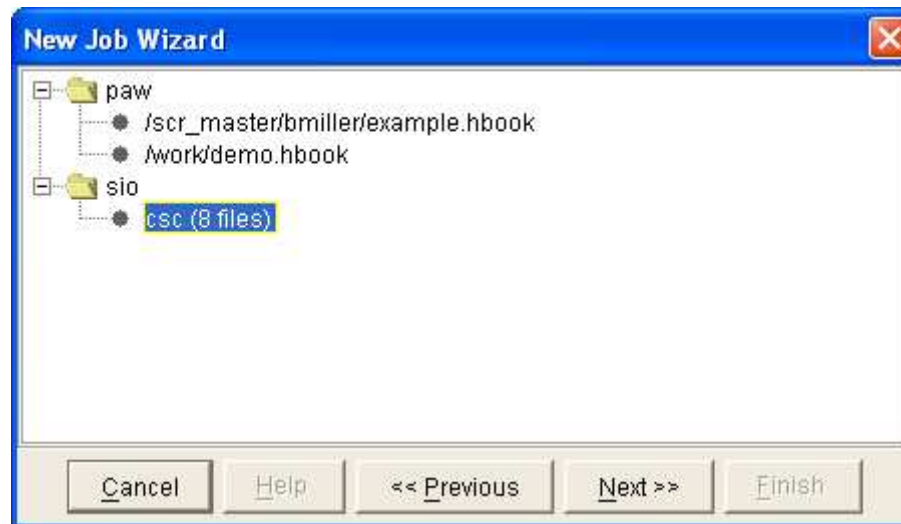
Two dialog boxes are overlaid on the main window:

- Grid Proxy Init:** A dialog box with a "Password:" field containing "*****", an "Options" button, a checkbox for "Use PKCS11 Device", and "Create" and "Cancel" buttons.
- New Job Wizard:** A wizard dialog box with the following fields and options:
 - Job name: "MyGridJob"
 - Job type: "Special Job" (selected), with a dropdown menu showing "Grid job".
 - Options for job type:
 - A local job for data analysis.
 - A remote job for data analysis.
 - A local job for generating events.
 - A remote job for generating events.
 - Buttons: "Cancel", "Help", "<< Previous", "Next >>", "Finish".



data selection

- Checks which data files are available to user
- System decides on which slave nodes to use.



grid enabled deployment

- Framework code is uploaded to slave nodes
- Data is moved or if on a mounted file system a link is created.
- Slave servers register with master server.





JAS client has Analysis Code Development Environment

Stanford
Linear
Accelerator
Center

The screenshot shows the Java Analysis Studio IDE. The main editor displays the following Java code for `CalEnergy.java`:

```
public void afterLastEvent()
{
    // Enter code here or delete method definition.
}

public void processEvent(EventData d)
{
    LCDEvent header = (LCDEvent) d;

    CalorimeterHits hits = header.getEMCalorimeterHits();
    double EMEnergy = sumEnergy(hits.getHits());

    histogram("EM nhits").fill(hits.getNHits());
    histogram("EM Energy").fill(EMEnergy);

    hits = header.getHADCalorimeterHits();
    double HADEnergy = sumEnergy(hits.getHits());
}
```

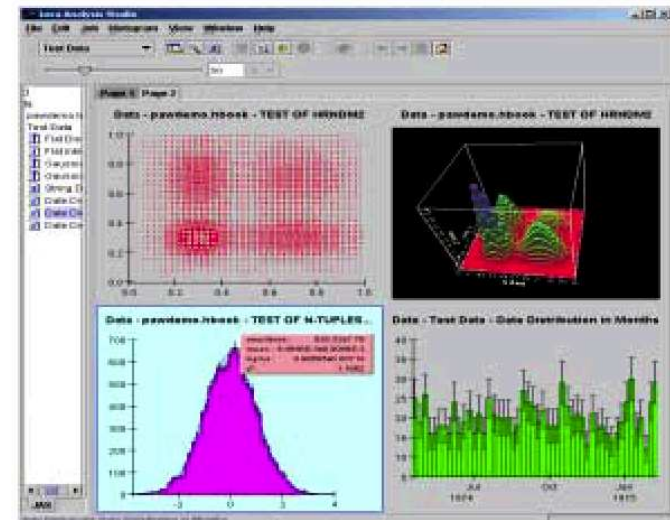
The IDE interface includes a menu bar (File, Edit, Job, Histogram, View, Window, Help), a toolbar with icons for file operations and execution, and a left-hand pane showing a project tree with folders like "Data" and "Test Data" containing various distribution types. At the bottom, a "Compiler Messages" window shows the output of a successful compilation: `javac C:\TonyAnalysis\CalEnergy.java` and `compile complete rc=0`. The status bar at the very bottom indicates "Line 24".



All JAS graphics capability can be use on grid jobs

Stanford
Linear
Accelerator
Center

- Rich histogram tools
- Re-binning slider
- Printing & Saving capabilities for publishing



Prototype was demonstrated at SC2003 from a Windows PC in the Phoenix Exhibit that connected to 16 nodes at SLAC.



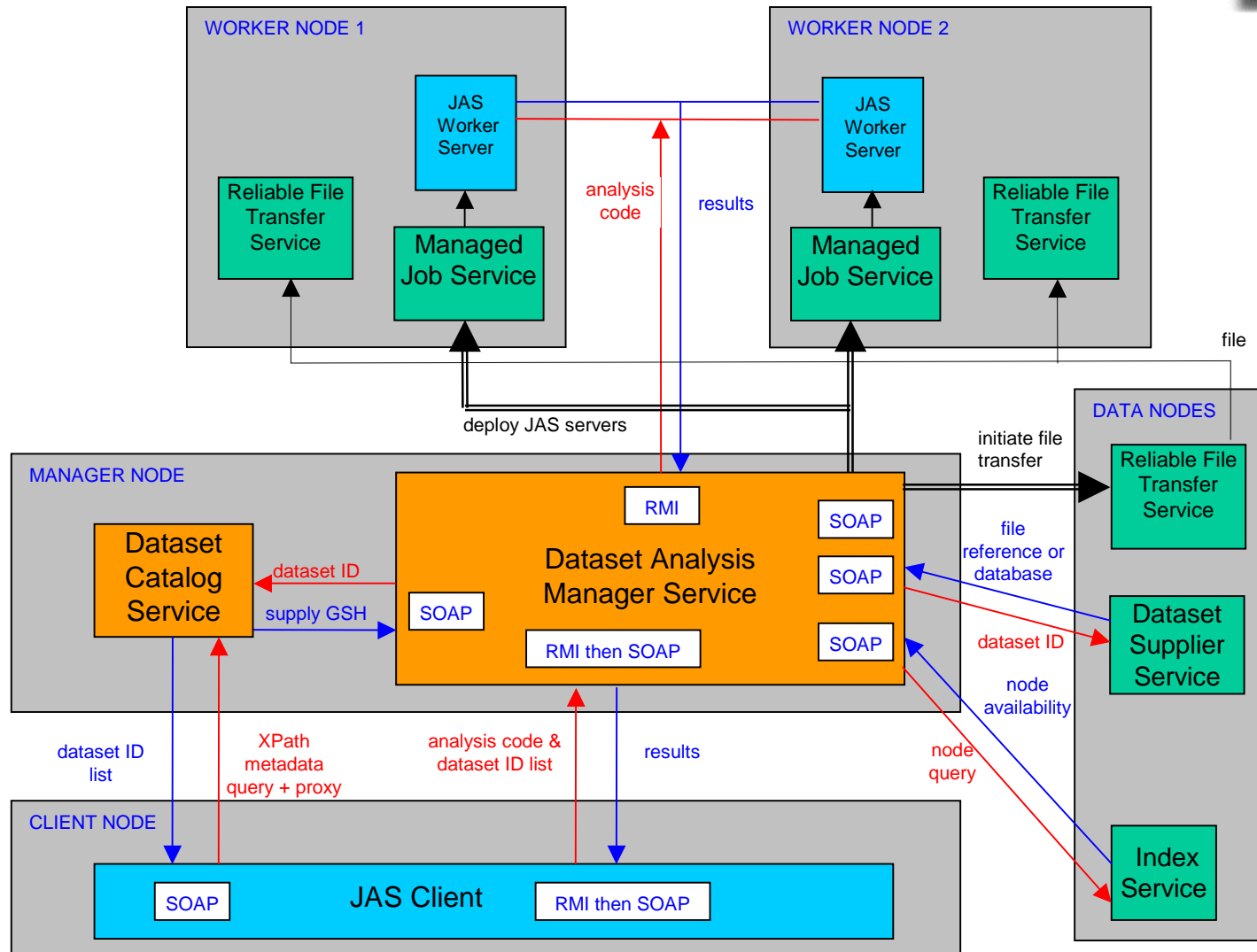
Phase II Design Goals



- Implements interfaces developed in Particle Physics Data Grid (PPDG) working group.
- Based as much as possible on OGSA style Grid services and to the maximum extent possible should use existing GT3 core services
 - Performance-critical parts may remain as Java-RMI connections
- Client will interact with a few key services which may connect to many back-end services.



Service and Communication Model for Phase II





We are Implementing PPDG Interfaces



- PPDG use case document written
- Collaborating with many PPDG members to facilitate interface reuse
- We will provide reference implementation
- Started with ***Dataset Catalog Service*** interface, meta-data can be browsed and searched

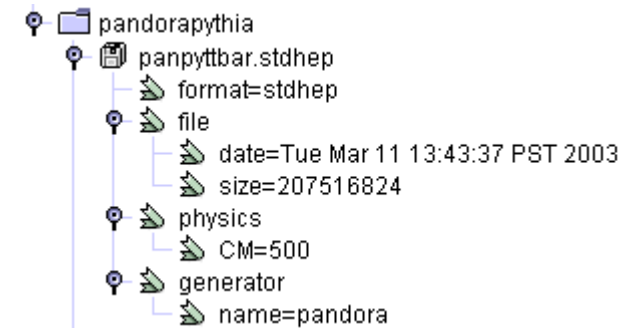


Dataset Catalog Service: Meta-Data and XPath



- Meta-Data

- No limitations on type or content of meta-data
 - At least initially meta-data is not typed (string)
 - Meta-data can be arranged into a hierarchy



- XPath Query Language

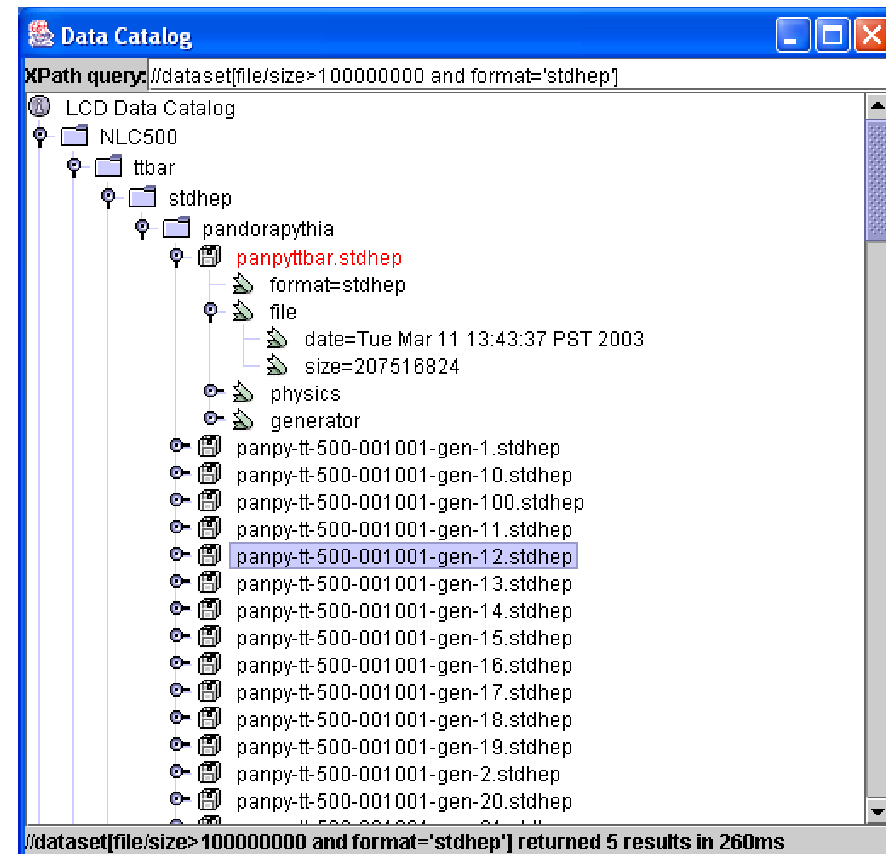
- XPath = language for querying hierarchical data structures.
 - Initially targeted at XML, but can be applied to any hierarchical data structure.
 - XPath = W3C standard, many implementations exist.
- Example Query:
 - `//dataset[file/size>100000000 and format='stdhep']`



Dataset Catalog Service: Reference Implementation

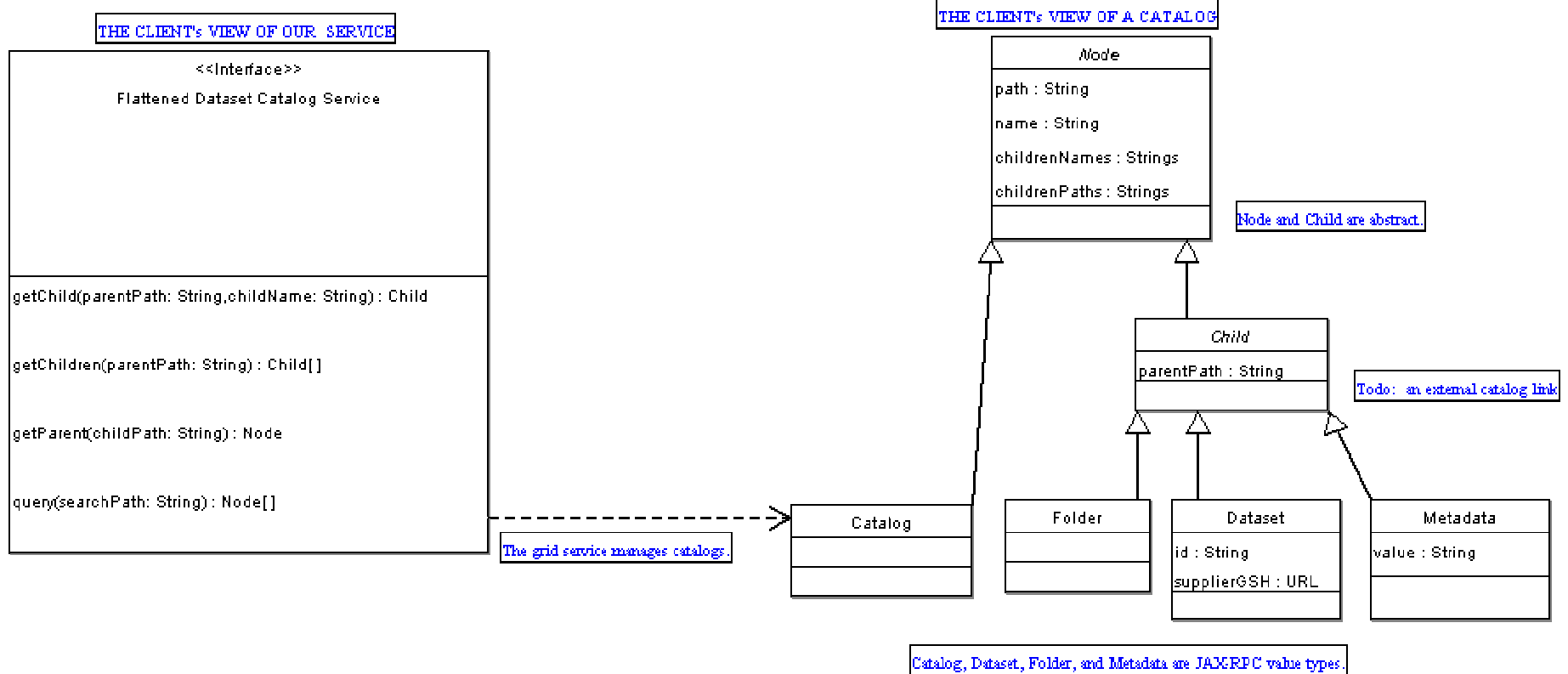


- Interface has been approved by PPDG
- Ref. implementation is only one possible implementation, could be implemented on top of any existing data catalog system.



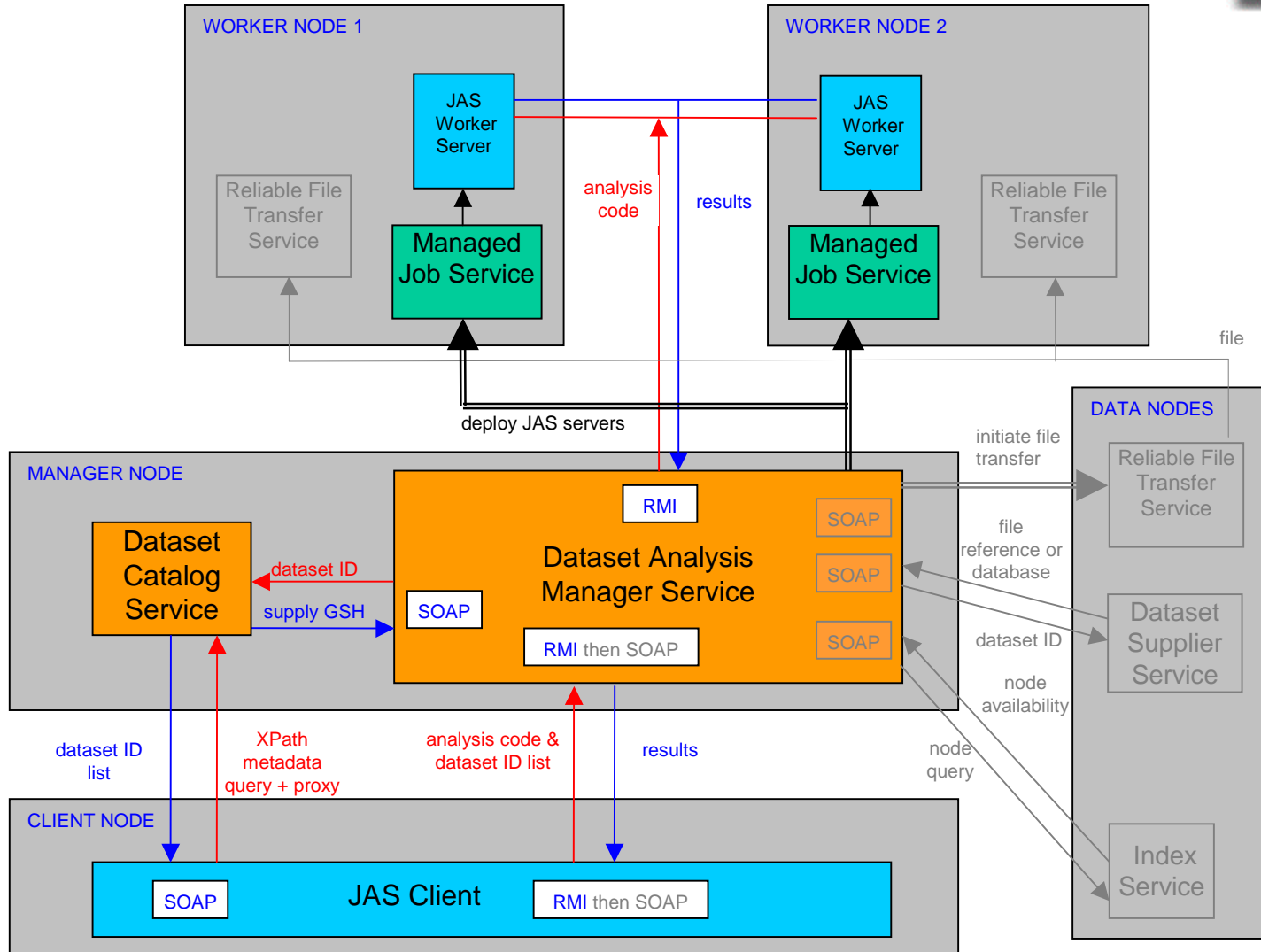


Dataset Catalog Service interface UML





Summary: Existing Implementation





Next Steps for Project



- Working on release version of system with Grid service (GT3) deployment
- Working currently with PPDG on Job Submission interface
- Add code to use RFT file transfer service when manager recognizes that data is not on mounted file system
- Move client from having RMI connection to using Grid service



<http://grid.txcorp.com>



Existing System Summary

