Proposal: MPI, Distributed Objects, Rendezvous, and Distributed Tasks and their use in Scientific Computing Part 1

Daniel Beatty

February 18, 2005

1 Feasibility Study

The three items that are the heart of using xGrid for both integrated Grid and general applications in my dissertation, and are suitable for the Spring semester 2005.

- xGrid
- Scientific Computing on the Wavelet Representation of the SDSS
- Wavelet Fits
- Rendezvous, Distributed Objects, Directory Services, NSThreads, and Distributed Tasks.

Issues of database relation optimization for the most part are going to be ignored. The distributable piece of this puzzle is the jocfits (Java-Objective C FITS) object is to be defined as an object containing either libraries or proxies to libraries for processing the image and its data. All other services will be either advertised objects(services), temporary objects generated by distributed tasks, or solely contained in the distributed task.

1.1 Evolutionary Direction

There exist an Application and Object Paradox whose nature submitted jobs such as numerical simulations. One example is in MPI programming. MPI uses a batch engine to launch many copies of same program which communicate with each other and use that communication to branch in paths to solve the problem the program was designed for. However, the what is the application. In many respects, the MPI batch engine could be seen as the actual application since it sets the work in motion.

Now consider the average application. Every bit of work is set in motion by the user operating on the application and the application calls any method necessary to fulfill the users wish. In MPI the environment is provide by the batch system. For general applications, the environment is provided by the operating system which generally has a much more sophisticated process manager.

If the desire is to maximize computational power, then a model needs to add a more techni-color spectrum to the idea of computational spectrum. To provide a technicolor federated scheme the federation must be able to discover its limits for each genre. The consequences to maximized computation with data includes:

- Discovery Services
- Look Up services
- Secure access to the data
- Confidence that proper services are being called.
- Accessible common file system
- Platform issues:
  - Platform Dependent Code
  - Non-platform dependent code such as JVM, .net, and intepreted languages.

1.2 SDSS Wavelet Application

This feasibility study uses the problems encountered by the Sloan Digital Sky Survey to show the need of this scheme. How does this scheme derive the use of wavelets? How does this scheme work? In fact, the application of wavelets to the SDSS data releases could be a topic in and of itself. However, there is a common need.

All of the SDSS data can be reduced to a collection of FITS files and the meta-data derived from it. These FITS files can be viewed in terms of objects where the data comes from the files, and the operations contained in the object make it compatible with conventional and wavelet numerical methods. Such an object can act as a proxy, and deliver the information as the transportable object itself.

What is the point of using wavelets in these FITS objects? First a matrix in a wavelet domain tends

to be more sparse and easier to compact. The wavelet domain also allows for feature detection, image enhancement, and noise reduction a much simpler operation. Can these operations be handled by these objects? Can these operations be handled concurrently? What message handling is in order? These are questions for the feasibility study.

1.3 Currently Available Distribution Schemes

1.3.1 Darwin and Distributed Objects

OSX is a product of NeXTStep with the Mach micro-kernel. As such it also has NSPorts. One feature that is also present is another service registration scheme called Rendezvous. Rendezvous is Apple's implementation of Zeroconf DNS which allows services to declare the name, type, port, etc.

OSX uses NSPorts to provide distributed objects (DO(s)) and uses the run loop and or thread to achieve a non-blocking solution. Such DO are called via normal message passing routines associated with Objective C and NS Objects. This mechanism provides a sort of proxy for which there are two classes" NSDistributedObject and NSProxy.

An NSConnection object has two instances of NSPort: one receives data and the other sends data. An NSPort is a superclass to all other ports. NSMachPort uses Mach messaging and is typically used solely on the machine itself. NSSocketPorts use socket to go between machines.

There are addition identifier/ modifier types applied to distributed objects: functions, methods and members alike. These key words are as follows:

- oneway void (client does not wait for a response.
- in (A receiver is going to read the value but not change it.)
- out (A value is changed by the receiver by not read)
- inout (receiver is to both read and write the value).
- bycopy (argument is archived before sent and de-archived in the receiver's process space)
- byref (the argument is represented by proxy).

Each connection can have a delegate. Each time the connection spawns a new "child" connection, the "child" will have its delegate outlet set to point to its parent delegate. The connection monitor is a class for logging delegates and their connections.

1.3.2 Distributed Tasks

Of course, there is nothing wrong with calling distributed tasks either. An example was provided by O'Reilly's articles and written by Drew McCormack May 11, 2004 [**?**]. This analysis examines the crucial parts.

Apply Filters is the method that calls Distributed Task. There are many nuggets of value in addition to the calls for:

- Add Sub Task with Identifier . This call includes
    1. The identifier
    2. Launch path

3. Working Directory
4. Output Directory
5. Standard Input
6. Standard Output

- Launch

The methods of how "Photo Industry" provides these values are somewhat interesting.

- The first section of Apply Filters acquires the time.
- Next initiates local instances of the file manager.
- The output directory is fed into Apply Filters and is not interesting.
- The temporary directory segment is interesting.
    1. It uses the processes own information (supplied by NS (OSX) which identifies the process in all of its details. The way this is used to access programs is with in the application itself.
    2. The temporary directory of functions which acquires the temporary directory as specified by the OS. (Any where NS applies).
- The next section claims to produce standard input for the filters which are actually programs and the parameters to those programs. The means for this is the typical array/ dictionary scheme of Objective-C.
- The next segment produces input and temporary directories for the input data (the photos). Features of these production(s) is the production of directories for the sub-tasks. Thus a scheme for dividing the work judiciously is being applied.

The question of the thread oriented submission becomes an issue.

Also, the feeding of data structures becomes an issue for the parent application:

- The manner the sub-tasks are divided up as a list of files (input). Items copied into these directories into these directories are the data (photos) and the programs to work on them.
- These structures include message forwarding which is the purpose of a NeXTStep delegate.
- "A delegate is an object directed to carry out an action by another object." page 456 [**?**]
- Once the sub-tasks and its data are determined, the sub job is copied out of the bundle (app), and the executable (script), then the sub-task queue is loaded.
- The rest of the methods are delegate methods.

1.3.3 Devise xGrid client from Source

One thing to be said about the distributed tasks devised by Drew McCormack is that it is a client of a client. Of course, xGrid has three basic components by design: client, agent and controller. Since xGrid's agent and client are open source, a good clean examination of these components may be in order to devise an xGrid API that makes any application simply a client of the xGrid system. Some of these clients could broker xGrid's services to Federated systems like SORCER, Condor, Globus, and the like.