Jim Gray
Distinguished Engineer and Manager
Microsoft Bay Area Research Center

Management introduction

Jim Gray is a Microsoft Distinguished Engineer. He is part of Microsoft's research group and is Manager of the Microsoft Bay Area Research Center. Over many years his work has focused on databases and transaction processing and he was awarded the ACM Turing Award for his work on transaction processing. He has also been active in building online databases like http://terraService.Net and http://skyserver.sdss.org.

In this discussion, Dr. Gray talks about his view of the Grid, as seen through a Microsoft lense. He describes:

- *how 'the Grid' is composed of multiple communities and interests*
- *the challenges that face each*
- *Web Services and OGSA and places these in context*
- *the commercial dimension.*

An informal Grid taxonomy

The Grid has at least five identifiable communities, each with a different interpretation of what it (the Grid) is about. These include:

- compute-centric users
- peer-to-peer advocates
- outsourcers
- data and application-centric advocates
- collaboration-centric (e.g., Access Grid) proponents.

The term 'Grid' was coined by compute-centric users led by Ian Foster and Carl Kesselman. Most of these people come from US government funded laboratories and super-computer centers like those at Urbana, Argonne, Fermi, San Diego, Livermore, and so on. The Grid book (I. Foster and C. Kesselman (Eds). *The Grid: Blueprint for a New Computing Infrastructure.* Morgan Kaufmann, 1999.) had chapters about collaboration, data, computation, and instrumentation. The community certainly understands the whole Grid spectrum; and is actively inovolved in projects that encompass data, telepresence, and portals. But much of the discussion here, and the work actually performed to date, focuses on running large computations (resource management) and moving the input and output files about (GridFTP).

This community typically runs big MPI batch programs. They operate huge machine clusters costing millions of dollars. Most just want access to some supercomputer somewhere. Some want to perform computations that are even larger than these clusters, so they need to connect these super-computer centers on a planetary scale. They envision the Grid as the sharing of compute resources.

One of the paradoxes of the supercomputing world is that SETI@Home delivers about 45 tera-flops of processing, all day every day. This is 30% more than the peak performance of the NEC Earth Simulator that claims to be the biggest computer. Indeed, SETI@Home is more powerful than the bottom half of the Top500 combined.

If you want cpu cycles, most of them are not in the data centers, they are on the desktops. Condor, Entropia, United Devices and others are harvesting these 'spare' cycles using peer-to-peer technologies – rather than the centralized computer-center model. The peer-to-peer advocates form an active and innovative branch of the Grid community. Broadly, they have a Napster, SETI@Home, or Gnutella heritage.

The peer-to-peer community wants to program the Web as a symmetric system of autonomous nodes; as opposed to a client-server architecture. Their model puts the processing at the periphery of the network. In addition they place most of their storage at the periphery. They are looking for applications that can harvest the capacity at the edge of network where the nodes operate under decentralized control and interact via symmetric protocols (rather than master-slave.)

Moving on to the next group, outsourcers, many have observed that the Grid is a resource. You send your problem to the resource; the resource sends back the answer.

Another name for this is outsourcing. The biggest outsourcing company in the world is IBM. IBM's Global Services is embracing and extending the Grid concept. IBM is one of the leaders in the Open Grid Service Architecture (OGSA) as a natural evolution of its outsourcing business. Other outsourcing companies are also climbing on this bandwagon.

Then there is the 'application and data centric' Grid subculture. I count myself in this group. These include people who believe that harvesting CPU cycles is not very interesting. That is not to say that compute intensive jobs do not exist; but most of these tend to be specialized in nature (like SETI@Home). Instead, this application and data centric group is more concerned about using applications and their data. Questions are sent to the application/data servers either because the data is huge (petabytes) or because it is encapsulated (proprietary). Application servers provide answers, while portals integrate the answers coming from multiple sources.

This application and data-centric group believes that what we are trending towards is a world in which:

- there are data centers, not computer centres
- most of the costs now lie in data management and networking, (while computing costs are less than sales tax.)
- you send requests to the data centers
- when requests arrive at multiple data centers, they face an interoperability problem.

To make everything work as desired, we need to have ways for:

- data to travel from one place to another
- intermediate results to travel from one place to another to be composed into answers.

This application Grid model is much more data-centric than compute-centric. It embraces all the OGSA, Web Services, .NET, SunOne, the peer-to-peer technologies, and anything else that works. Its goal is to make it easy to:

- program the Web (?) to access the data in it,
- send not just bytes but information (objects and the methods associated with objects) around the network.

The current plan is to build the Data Grid using OGSA which is in turn based on Web Services, XML and SOAP.

What is significant here is that SOAP really is a Simple Object Access Protocol. The layers that are built above it really are an object model. You could say this is 'Internet-scale CORBA' that really works.

The collaboration-centric 'Access Grid' community forms the fifth Grid community. It wants to use the Grid network bandwidth to allow high-quality person-to-person communication.

The Access Grid is a teleconferencing and distributed meeting software platform that enables people to minimize travel while still communicating effectively. To achieve this, the Access Grid needs copious bandwidth which means it needs a very high speed network -- which is one deliverable that the Grid is likely to put in place.

Indeed, Gordon Bell asserts that the Access Grid will be the most significant out-growth of the Grid, much as Mosaic (the browser) was the main social benefit of the previous supercomputing generation. That's silly!

These multifaceted 'interpretations' of the Grid are good; there is something in it for everyone. If they look, most people will find something, in it that they really like. The Grid is a big tent in which most everyone can find their dreams.

The major challenges and reality

The major challenges depend on where in the big tent you happen to be standing. The Access Grid is a fascinating research area. Right now the software and hardware base is chauffeur driven: you have special room managed by an Access Grid specialist (to set up the four different linkages and operate the four different computers and many different applications).

If you participate in an Access Grid event there is a good chance that some piece of the technology will not at the right moment. As such, the Access Grid is fragile and its architecture needs two more prototype generations before it will become useful to you and me.

Nevertheless, I think the probability of success of the Access Grid is certain. This is why Gordon Bell is so enthusiastic about it. Microsoft's ConferenceXP is our shot at the next generation and is receiving good reviews – it is easy to use and can be inexpensively set up in your office or in a small conference room. It is a one computer, lower bandwidth, and lower cost version of the Access Grid. The office/laptop version runs on your current system. A classroom is about US$25,000 and is self-managing. We are working actively on

trying to bring down both the cost and the complexity of having an Access Grid node.

The peer to peer guys continue to have just extraordinary enthusiasm. In part I think this is because their approach appeals to people's democratic and egalitarian notions -- which everybody can contribute to the computer in the sky. Napster was as much a social movement as anything else: students had Napster decals on their dorm room windows.

To me, the peer to peer community is robust. The challenge it faces is finding applications. File sharing was one application -- sharing of music or recipes and so on. **SETI@Home** is also a kind of peer to peer application and there were companies founded to ride the peer to peer revolution (although most of those companies are now gone.)

The peer to peer space is competitive. It is not an easy space in which to establish a profitable business model. Both United Devices and Entropia have found their best customers to be large organizations that want to harvest the spare cycles within their own enterprise.

This within-the-enterprise cycle harvesting is a pretty narrow form of peer to peer. But it is still relevant. One observation that a customer made to me recently was that the likes of the US Government, AT&T, Ford and GM already have substantial internal networks. In fact they are provisioning their networks with gigabit backbones. They are already in a position to build an 'intra-Grid' within their organization. Indeed, I believe that United Devices and Entropia are having their biggest successes by building such intra-Grids for harvesting these spare CPU cycles and supplying these to others within the same organization who are cycle-limited rather than I/O limited.

The peer to peer guys are now coming out of the manic phase when there was such incredible enthusiasm and hype. They are now in the phase of having to deliver. The challenge they face is finding more applications that can exploit the edge nodes of the Internet and Intranet.

So, moving on to the next Grid group, the application and data centric group is the area where I have been working. It has a great deal of velocity. In Microsoft, the .NET guys are beavering away. There are several thousand developers in Redmond actively programming to build .NET and really exploiting the emerging GXA stack of advanced Web Services. There are similar efforts at IBM, Oracle, BEA and other large software houses.

And of course, there are the Physics, Biology, Medical, Earth Sciences, Chemistry, and other scientific disciplines that are embracing the grid as a way to publish and access scientific data. This is happening in Asia, Europe, and the Americase.

This is the area which is also most familiar to those interested in middleware. It is where conventional data processing meets the Grid and addresses:

- naming
- security
- authorization
- data representation and interchange
- programming models
- transactions
- co-ordination
- work flow
- etc.

To give some idea of progress, there are about 25 Web Services specifications in the pipeline, having to do with anything from queuing to work flow to how to represent data, schemas, query languages, etc (Microsoft calls this GXA for Global XML Web Services Architecture). The volume of activity is immense. It is impossible to stay current with all the things that are going on. There is just so much happening.

I know it is Microsoft's main focus, and I believe other software companies are moving in the same direction. Sun's recent joining of Web Services Interoperability (WS-I) seems to have ended the final hold-out.

It is easy to discount this, to say 'I've seen this phenomenon before, this is OSI or … all over again' -- where many people met to agree standards but few were interested in implementing many of them'. I think this is different, but only time will tell. I see two major differences.

There difference I see is that people are implementing in parallel to the definition of the specifications. Some leading-edge customers already in production – that's different.

The second major difference from past efforts is that the specs are pretty simple and they start with issues that people care about -- like how to:

- represent data

- represent queries
- provide security.

.NET was a huge gamble on Microsoft's part. It thought that XML was different. Around 1997, it redirected a large part of the company towards XML Web Services. .NET came out around 2000, after three years of engineering. Today we are five years down the road and it looks like a pretty good gamble. The parts are coming together.

Let me illustrate why with a story. I was at a customer event recently. There was a panel about Web Services. The panel was chaired by a pundit who started by saying that this was 'just old wine in a new bottle', that there was 'nothing new or good here' and that it was 'all hype'.

The next speaker was the CIO who said 'Well, yes and no. This company is a J2EE enclave, using WebSphere and WebLogic. We could not get WebSphere and WebLogic elements to talk to each other through RMI. So, rather than trying to talk in the same language (Java) we decided to talk in a different language -- Web Services. It worked. The reality is that it remains difficult to get two different RMI implementations to talk to each other.'

This CIO went to describe how his organization had some ten major projects under way using Web Services. Three of these had been deployed and two are in beta. The resultant systems are much more maintainable, much less fragile (than the binary based systems they had worked with before) -- primarily because his people could actually look at what was crossing the wire with tools that already exist.

Web Services really do focus on interoperability. They pay a huge price for going through XML, but the benefit is interoperability. Sending you a floating point number through XML is not pretty – but it is very easy to make work at both ends of the wire. The relevance to the Grid is immense.

That said, there has been a huge amount of hype about Web Services. I suspect that there are a fair number of people who have no idea what they are. When they find out, they will be disappointed that they are so simple. For middleware experts, Web Services can be described as 'XML meets CICS' or 'XML meets LU6.2' or 'XML meets CORBA'. That was what the pundit meant when he said new wine in an old bottle.

So the challenge here, and this is really a challenge for IBM and Microsoft, is how seriously each company will take the Web Services Interoperability consortia (WS-I). Interoperability is critical. The need is for them to agree that interoperability is more important than going out to build a better system than is available from competitors. The fact that Sun finally joined WS-I suggests that no one can now ignore it.

This brings us back to the Grid. The Grid may yet turn out to be a force for good -- by keeping vendors true to interoperability as the critical need. In addition, the outsourcers will need this as well if they are to have a business

Now let me turn to the progenitors of the Grid -- the compute-centric people. Everybody is hopping onto the Grid bandwagon. But, what is worse for this group is that computing is becoming free. It is much easier to deal with your local Beowolf cluster than it is to deal with some remote computer center. The reason people built Beowulf clusters is they could not stand to deal with super computer centers. In the end somebody has to pay.

The economic model for the traditional super computer group no longer works. It is too painful and there is too much overhead associated with getting cpu cycles from some high-overhead computer center. The only reason to go to a remote facility is that it may have something you want (like data or information).

That said, the compute-centric Grid community is working to build batch job schedulers to run huge MPI programs on large clusters. This seems retro to those of us who spent our careers replacing

- batch programming with timesharing**, and**
- batch transaction processing with online transaction processing**, and**
- batch data analysis with online analytical processing.

But, I guess batch is no longer retro – it is the wave of the future.

This Internet scale Beowulf faces a problem though: it requires problems that have huge instruction density. You can characterize a job by:

- how much I/O it does**,**
- how many instructions it uses
- how much network traffic it requires.

Right now a dollar buys

- 1GB of network traffic

- one day of CPU time
- 10 days of disk arm time.

If you look at SETI@Home or problems involving protein folding, they do about a million instructions per byte of data. Given that high instruction density, it is economic to send the data to someone else for processing (buying network time is cheaper than buying the processors and performing the job locally). But most of the jobs I see have an instruction density of 10 instructions per byte to 10,000 instructions per byte. In those computations it is much cheaper to send the program to the data. For those jobs the CPU cost a tiny fraction of the data access and network transport cost.

So I think that the compute-centric guys need to move towards the data centric model. This is not a huge issue; all they have to do is embrace and extend the data center model and they will be done. Indeed that is what OGSA is about. But it does move the focus away from batch job scheduling.

OGSA and Web Services

Right now OGSA is mostly wrapping Globus in Web Services -- translating each Globus service to an XML/SOAP equivalent. This is a good first step. But the question that this raises is: what are the Web Services that arise when you have a strong object and data model?

The compute-centric folks are very bright and very hard-working. They are genuinely trying to make progress. I believe they recognize the data modelling and object modelling problem and are working hard to address it.

Paradoxically Tony Storey (of IBM) and Malcolm Atkinson (of Edinburgh University) as well as various other people in the European data Grid are much stronger in the area of data representation. Something I find surprising is that Europe is likely to lead the definition of:

- what a data set is
- how data is represented
- how XML data fits with the Grid.

This is borne out in other ways. This is understandable. Its financial support is different. Tony Hey is concentrating on e-Science and e-commerce. Necessarily, that makes the UK's Grid activities much more focussed on data and business possibilities.

Commerce and the Grid

Any large organization that has inter-operability problems is going to love Web Services. Web Services are becoming the base for the data-centric view of the Grid. Ergo, commerce should love the Grid -- even if it has yet to discover this, especially as the Grid logically extends the interoperability available within an enterprise to working between two or more enterprises.

That is not to say that all is done. Web Services today are mostly about inter-operation between one part of an organization and another. They usually do not require high performance. But high-performance will come with time.

While it is fair to say that XML performance remains dismal, this is little different to the early days of HTML. There are efforts to deliver high-performance XML at Microsoft and other companies. The Microsoft model is that we will have proprietary implementations of these open standards. People will buy one implementation over another based on TCO – productivity, functionality and performance. This is what distinguishes the various SQL systems today.

For example, people are now doing TPC benchmarks using .NET. tpcC benchmarks using Web Services (rather than HTTP/COM+) in fact show improved performance.

Now think of the astronomers. I have been working on building a worldwide telescope. In the past astronomers had lots of telescopes and lots of data coming from those telescopes but few means to pull the data together in meaningful ways. That has changed. What was hard before .NET (and Web Services) is easy now. Most of this has been about solving inter-operability issues -- essentially business to business communication or, in this case, astronomer to astronomer communication.

So I see the Grid as an engine of change engine, even of discontinuity. This is complicated. You can emphasise the similarities or the differences.

The Grid will consist of at least my five 'views'. If you throw .NET and Web Services into the Grid, it is much more significant than it used to be. But what really matters for business is that it (business) adopts the view that is most appropriate to its needs. Each business needs to look at these many aspects and see what parts of this pie actually help – rather than just blindly embracing peer-to-peer

or Globus or Access Grid because everyone else is doing it (I see a lot of this).

That said, there is a business dimension that I do not think many have considered. Much of the Grid discussion occurs in an economic vacuum – everything is free. While Web Service applications do not need very much bandwidth, many of the Grid applications seem to need huge bandwidth capacity – because they are moving data around the network.

The problem is that such bandwidth is not there at an affordable price. The instruction density has to be at least a million instructions per byte of network traffic given current economic pricing. There are very few applications that match this profile. This is a real barrier.

In addition, there really are cultural differences between scientific applications people and business applications people. We are trying to bridge these barriers. But this is difficult when the scientific guys fundamentally do not have the experience with structured data and they have an MPI computational model. They are much more oriented to files than databases.

Nevertheless, I think many tools out there are applicable to the each various individual views of Grid computing. These are going to change many aspects of computing. For example:

- peer to peer is finding niches in Monte Carlo simulations**,**
- the uptake of .NET and Web Services for building the data Grid is huge**,**
- the Access Grid, if Gordon Bell is right, will likely overshadow all these developments.

Management conclusion

Dr. Gray has been at the forefront of both transaction processing and databases advances over the past 30 years. He is in an excellent position to observe the evolution of the Grid, and to categorize its many interested parties.

In his discussion, three points stand out:

- *the importance of Web Services, however named*
- *the reality that the Grid is already supporting different interests with different objectives*
- *the likelihood that the Grid will change the commercial world's way of operating, once it realizes what the Grid can deliver.*