



I would like to state up front, that I wrote this article because I lost a bet. No money changed hands. The bet was just a gentleman's agreement with myself that I would have to do an article on the subject of the winner's choice if a military organization ever produced a functional workflow system that also met federal records management requirements. Thanks to Navy Lt. Jamie Gateau and his demonstration of the Prototype ERM (Electronic Records Management) Implementation at Naval Network and Space Operations Command (NNSOC), this issue's edition of the Lazy Person's Guide will address grid computing.

The term "grid computing" came into fashion in the mid-1990s to describe a wide variety of distributed computing projects. The term itself brings to mind an interconnected grid of machines that form a single, massive entity. Reform that vision slightly to account for the fact that the "grid" can be any shape, can cover the entire world, and consist of computing devices that can range in size from desktop PCs (personal computers) to large mainframes, and you should get a good mental picture of a global grid system.

However, grid computing involves more than just the physical architecture that comprises its network. It also includes all the associated information resources and the protocols used to make dozens, hundreds or thousands of disparate devices essentially function as a single unit. Properly implemented, a grid could become the organizational equivalent of an intelligent, nominally self-aware technological central nervous system that provides access to and coordination of computing power, data, information and knowledge for an organization by efficiently using every available resource all the time.

Divide and Conquer

Before we get to a more detailed look at grid computing, let's first look at its most famous predecessor: *distributed computing*, which also involves a number of devices each sharing a single purpose. However, distributed computing is a much simpler basic concept: Break up an activity that consists of many similar, repetitive tasks and distribute those tasks to every available machine for processing. Here's a simple example of how to construct a distributed computing exercise. Let's say we want to build a multiplication table listing all the multiples of positive integers from 1 through 12 (e.g., 1x1, 1x2 through 12x12). Including duplicates (like 3x5 and 5x3), there are 144 calculations. The application controlling the distribution then sends each calculation, one at a time, to each of the machines con-

tributing processor time. So, 1x1 is sent to machine A, 1x2 is sent to machine B, etc., until all the available processors are working. As each machine completes its task, it sends the result back and is given a new one.

Probably the most famous example of distributed computing is the Search for Extraterrestrial Intelligence (SETI) project. SETI is a scientific effort to determine if there is intelligent life outside Earth, primarily by using the Arecibo Observatory radio telescope in Puerto Rico to detect artificial radio signals coming from other stars. However, the telescope was collecting more data (about 35 gigabytes per day) than SETI could process by using its own systems. They could not afford a multimillion dollar supercomputer and processing on the systems they could afford would take decades. It is a problem that anyone suffering from information overload can sympathize with: Too much information, not enough resources to process it.

Then someone on the SETI team looked around the office, saw toasters flying across every idle desktop computer in the room, and got a brilliant idea: Develop a screen saver that would process SETI data. Distribute that screensaver to millions of users around the world and let it use spare clock cycles to process packets of SETI data. Thus was born SETI@home which lets anyone with a computer and an Internet connection participate in the SETI project simply by loading a distributed computing client on his or her home computer to analyze packages of data collected by the Arecibo Observatory. Because all of the packages can be processed independently, much like our multiplication table example, the SETI project is a perfect match for a distributed computing application.

Consider that there are over 5 million registered SETI@home users, some with more than one computer running the SETI screensaver. An average desktop computer with a 1-gigahertz central processing unit (CPU) can process 1 billion floating-point operations per second (flops), or 1 gigaflops. The world's most powerful supercomputer, Japan's Earth Simulator, runs at 35 teraflops (or 35 trillion floating point operations per second). As of July 2004, the SETI distributed computing project is running at about 14 teraflops, which would easily make it one of the top 5 supercomputers in the world if it were a single system and teraflops were the only measure that counted. It's an impressive amount of computing power all dedicated to helping search the skies for signs of intelligent life in the universe.

Breaking encryption is another use for distributed computing. It was a distributed computing project that originally cracked the National Security Agency's 56-bit digital encryption standard (DES) in 30 days. On the next try, it was done in 30 hours. On the last attempt it only took 3 hours, which is impressive if you consider that 56-bit DES was once considered secure enough for most Internet transactions. Part of the speed increase has been attributed to faster processors, but an equal measure of the improvement was due to better management of distributed computing resources.

The Dark Side of Distribution

In addition to SETI's rather benign use of distributed computing, there have been some less friendly applications. Chief among these have been distributed denial of service (DDoS) malware (malicious software) spread through various means that infects a large number of computers and then uses them as attack platforms to flood Internet servers with more traffic than they can handle.

More insidious use of distributed computing is spyware, software that loads itself onto your computer and then reports on what you do to some central database. Most spyware is commercially motivated. Marketers are simply sampling to better target their advertising, goods and services. However, there are some spyware applications that will steal data from your computer. Credit card numbers and activation codes used during online shopping are a prime target for this type of distributed malware.

A less nasty but still annoying form of distributed application is adware. Once installed, adware will pop up or insert advertisements on your screen. Some adware actually comes bundled with commercial software programs. I highly recommend that you read every line of every end-user license agreement (EULA) that comes with every piece of software you buy. Pay particular attention to anything that reads, "...and we reserve the right to install software on your computer that will periodically send information back to us." Starting with the stage.dat file incidents 12 years ago where the Prodigy consumer information service was discovered sampling subscribers' hard drives, various companies have tried many methods over the years to grab as much data from PCs as possible without either getting caught or arousing the ire of their customers. Given the potential information bonanza, can foreign intelligence agencies be far behind?

Less capable than spyware and adware, but still in the same class are the cookies you pick up while visiting various Web sites. Cookies are small files associated with particular Web sites that store information about your interaction with the site. Sometimes they store more than that. Cookies come from two main sources.

First-party cookies come from any Web site you visit directly. They can either be long-term cookies that are intended to reside on your computer for several years or session cookies that only track what you do during your current interaction with a site. Long-term cookies are usually used to store information like ID and password for automatic login, personal data that identifies you to the site owner and other information collected through your interaction with the site that may be of use to either you or the site owner. Session cookies are usually used for information with less long-term value, like what's in your current shopping cart at an online shopping site.

Third-party cookies come from sites other than the ones you visit directly. Visiting a commercial news site like CNN or *The Washington Post* Online will load their cookies, but you may also get cookies from advertising sites like HitBox or DoubleClick that have agreements with the host sites. Third-party cookies are of no benefit unless you like advertisers knowing your shopping demographics. If you want to shut them out: From Internet Explorer, go to Tools/Internet Options/Privacy/Advanced and set the third-party cookies option to "Block."

If you're concerned about spyware, adware or cookies, I recommend a couple of free programs that may help: Spybot S&D (for "search and destroy") and Ad-Aware v6. Both are freely available from <http://www.spybot.info> and <http://www.lavasoftusa.com> respectively.

Bear in mind that all this information harvesting is not limited to the Internet and World Wide Web. It all started with the introduction of the bar code scanner in the retail commercial world. Every

product purchased and scanned goes into some retailer's database, which then mines the data to see what, how much, when and where people are buying. If you use a credit or debit card of some type, companies can develop detailed individual buying profiles on individual customers.

Let's take a look at what separates modern grid computing from other forms of distributed computing. While distributed computing resembles an anthill, with all the little drones working to support the queen, grid computing is a more communal system with complex resource and rights management issues. It is the next order of magnitude in networking.

Enter the Grid

There are a wide variety of opinions as to what qualifies as grid computing and some people with a project to push or a product to sell will slap a grid label on whatever they have to offer. However, I did find one reference that I believe covers all the bases: "*The Anatomy of the Grid: Enabling Scalable Virtual Organizations*," by Ian Foster of the Argonne National Laboratory and the University of Southern California (USC), Carl Kesselman of the University of Chicago, and Steven Tuecke from USC, in the International Journal of Supercomputer Applications in 2001.

While not the most recent document I have seen on the subject, it is a comprehensive and detailed description of what a grid should be. I will attempt to quote and summarize the main points here, but if you are interested in grid computing on any level I highly recommend you read the entire paper, which is available on the Web at <http://www.globus.org/research/papers/anatomy.pdf>. In addition, the Globus Alliance Web site (<http://www.globus.org>) is the single most comprehensive collection of objective information on grid computing I have found. Here are a few of the key points from the paper:

⇒ *"The real and specific problem that underlies the grid concept is coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations."*

Traditional networks tie together computing resources so they can communicate but not necessarily cooperate. All the various networked personal computers on a LAN use universal services like file storage, networked printers or access to an e-mail server. But if your desktop computer runs out of processing power for whatever it's trying to do on a traditional LAN it can not call on your neighbor's desktop (or one 1,000 miles away) to pick up the extra work. That capability, however, is a goal of the grid.

⇒ *"Because of their focus on dynamic, cross-organizational sharing, grid technologies complement rather than compete with existing distributed computing technologies."*

Intranets tie machines together, network storage provides repositories for data and information, and system inventories and file indexes list available resources. Grid technologies provide the next level of evolution by replacing the "static configurations" associated with connected but stand-alone processors and storage forming a dynamic, shared pool of resources. A grid can allow remote access to information, applications, sensors, etc., that previously were only available through dedicated systems.

⇒ *“Resource sharing is conditional: Each resource owner makes resources available, subject to constraints on when, where and what can be done.”*

This will keep someone else’s computations from taking over every clock cycle on the grid. Beware of limits on access to data and information because the whole point of a grid is to facilitate sharing.

⇒ *“In defining a grid architecture, we start from the perspective that effective [virtual organization] operation requires that we be able to establish sharing relationships among any potential participants.”*

This means interoperability, interoperability and interoperability. It is absolutely essential that a grid, as with any networked system, be based on common operational protocols, services and application programming interfaces.

Foster, Kesselman, and Tuecke also describe a model of grid architecture consisting of the following five layers:

• *“The grid fabric layer provides the resources to which shared access is mediated by grid protocols: for example, computational resources, storage systems, catalogs, network resources and sensors.”*

• *“The connectivity layer defines core communication and authentication protocols required for grid-specific network transactions. Communication protocols enable the exchange of data between fabric layer resources. Authentication protocols build on communication services to provide cryptographically secure mechanisms for verifying the identity of users and resources.”*

• *“The resource layer builds on the connectivity layer communication and authentication protocols to define protocols (and Application Programming Interfaces (APIs) and Software Development Kits (SDKs)) for the secure negotiation, initiation, monitoring, control, accounting and payment of sharing operations on individual resources.”*

• *“While the resource layer is focused on interactions with a single resource, the next layer in the architecture contains protocols and services (and APIs and SDKs) that are not associated with any one specific resource but rather are global in nature and capture interactions across collections of resources. For this reason, we refer to the next layer of the architecture as the collective layer.”*

• *“The final layer in our grid architecture comprises the user applications that operate within a [virtual organization] environment.”*

The authors also include discussion of other issues, including dealing with user authentication; single sign-on; integrating application and storage; and enterprise and peer-to-peer technologies within a grid. And finally, in a very useful section, they describe that a grid:

⇒ *Is not a next-generation Internet, but a set of services and applications that enhance the connectivity the Internet provides.*

⇒ *Is not a source of “free” cycles. Grids enable “controlled sharing,” not unlimited access to everyone else’s stuff.*

⇒ *Does not require a monolithic distributed operating system, but should instead follow the Internet Protocol model of open standards.*

⇒ *Does not require new programming models, as the challenges of building a grid are not fundamentally different from those already encountered in traditional networking.*

⇒ *Does not make high-performance supercomputers superfluous. They will still be needed for computational problems requiring low latency and high bandwidth. The authors suggest that grid computing may actually help increase demand for them by allowing more participants to tap their resources remotely.*

Updates and Final Words

In the CHIPS Summer 2003 issue I reported that Apple Computer had developed a zero-configuration networking technology called Rendezvous. In a development that I believe will play a role in grid development, they have now released a Rendezvous developer’s toolkit for Windows, Linux, BSD and Java. You can find more news on the Web at <http://maccentral.macworld.com/news/2004/06/30/rendezvous/index.php>. (Due to resolution of a trademark dispute, Apple has now changed the name from Rendezvous to “OpenTalk.”)

For those of you whose grids may eventually include wireless networking, you may be interested in the recently released 802.11i wireless Ethernet security standard. The best link I have found is at Wikipedia at http://en.wikipedia.org/wiki/IEEE_802.11b, a free Web-based encyclopedia that is becoming one of my favorite sites on the Web.

It has been said that we humans only consciously use, at most, 15 percent of our thinking power. Perhaps that is why our networks, at least compared to the potential we have seen for grid computing, seem to follow suit. However, if you already have or are building the physical structure necessary to support a grid (i.e., an organizational intranet), planning for the evolution to a grid before the intranet is set in stone is a good idea.

Well, I hope I settled my debt. Thanks again to Lt. Gateau and NNSOC for the inspiration. I look forward to hearing about their grid project some day — or maybe writing about it!

Until next time, Happy Networking!

Long is a retired Air Force communications officer who has written regularly for CHIPS since 1993. He holds a Master of Science degree in information resource management from the Air Force Institute of Technology. He is currently serving as a telecommunications manager in the U.S. Department of Homeland Security.

CHIPS



Don't miss a single issue of CHIPS, please send address changes to chips@navy.mil.