



HP's Hidden Supercomputers

- A research paper on Peer-2-Peer Computing

Infrastructure Strategic Engineering

- Operating Systems Group
Bill Dyck,
Yan-Fa Li
Michael C Smith



**INFRASTRUCTURE
STRATEGIC
ENGINEERING (ISE)**

Version 1.3 (HPWorld)

Copyright © Hewlett-Packard
Company 2000.

This document contains information and conjecture that is protected by copyright. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, transmitted, adapted, or translated, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without prior written permission, except as allowed under copyright laws.

Table of Contents

Abstract	3
Introduction.....	3
Case Study: Napster	4
Peer-To-Peer Computing	4
How Idle Is Your Computer?.....	6
Pioneers in Peer-to-Peer Computing.....	6
Peer-to-Peer Computing in the Spotlight.....	7
The Coming P2PC Explosion	7
Industry Attention and Standards.....	7
The Social Benefits of P2P	8
P2PC Application Models.....	8
United We Stand, Together We Process	8
Network Power	9
Gremlins, Saboteurs and Crackers	10
Peer-to-Peer Computing Business Models	10
Model #1: Brokering.....	11
Challenges: Auditing and Accounting	11
Customer Billing.....	12
Paying Suppliers	12
Model #2: Build your own with our tools.....	12
Traditional Development Funding Model	13
What areas could P2P Computing impact?.....	13
What Security Issues Exist with P2P Computing?	14
References.....	15
WWW Pages For Companies and Organizations Mentioned.....	15
Additional WWW Pages for Additional Organizations involved in P2P Computing	15
Appendix A – Processor Performance Comparisons.....	16
Benchmarking Processors.....	16
Peer-to-Peer Computing Equivalency Formula	17
Example of Comparison between Processors	17
Pentium II Intermediate Measurements	18
Pentium III Intermediate Measurements.....	18
Final Results.....	18

HP's Hidden Supercomputers

Abstract

Peer to Peer, (P2P), has become the latest buzzword in high performance computing circles. From the notorious Napster music sharing controversy to the Search for Extra Terrestrial Intelligence (SETI) Program, the P2P paradigm is here to stay and actively being used in dozens of projects, both commercial and altruistic, to solve hard and time consuming problems.

P2P can be defined as many individual computers connected via networks sharing their computing resources. Applications have included solving problems that previously required access to a super computer and, more recently informal sharing of storage space to distribute arbitrary files, most notably music and video in the form of MPEG formats (Moving Picture Experts Group).

The big attractions of P2P are the low costs of entry, and the ability to harness previously untapped resources hiding within many companies, and potentially individuals, computing infrastructures.

This paper is primarily about P2P Computing. It hopes to explore it's ramifications and theory and then ask questions like:

- What kinds of P2P applications exist today ?
- Who's benefiting from P2P ?
- What business models have proven successful ?
- How can HP take advantage of P2P ?

Introduction

Twenty years ago something like P2P would have been unthinkable. Ten years ago it would have been unlikely. Today it is a reality. Cheap processing power, thanks to the PC revolution, combined with access to the Internet have created a fertile ground for this new computing paradigm to exist within. Current P2P technologies can be classified under two broad categories: Computing and File Sharing.

P2P Computing can be defined as, idle CPU cycles contributed to computationally challenging problems using specialized software clients. Special servers, distributed across the Internet, partition the problem space into bite-sized chunks. Clients periodically check in with these servers, uploading previously completed units and downloading new ones.

Typically each client is identical, and can operate in a disconnected mode of operation, i.e. it isn't necessary to be online all the time. Successful projects support a wide variety of operating systems, including Windows in all it's incarnations, Unix (Linux, BSD, Solaris, HPUX, etc...), MacOS and even less popular OSs like BeOS.

File Sharing is what it claims to be. In today's conventional infrastructure, files are placed on centralized FTP or Web servers. In a P2P model, files are stored all over the "network". P2P clients typically operate in a dual client/server mode, offering both capabilities within the same application. By connecting to the P2P "network" you are



adding your resources to the pool in real-time and become part of the “network”.

So in practical terms, what is P2P ? We’ve already put forth a more academic definition, so let’s look at an existing application and examine how it works to give us a sense of scope for other potential uses.

Case Study: Napster

Just about everyone who listens to music is probably aware of the company [Napster](#) and the product it provides.

Napster was the creation of a young North Eastern freshman named Shawn Fanning. Fanning had a great idea. Tired of hunting around looking for MP3 music files in newsgroups or on secret ftp servers, he would create a service where people who were interested in the same kinds of music could find each other online, building communities and share their files with each other directly off their computers.

Napster’s technology allows people to link their computers together into an enormous shared file store, purely for the purposes of sharing music. In a process known as ripping, individuals convert music from CDs, into compressed MP3 versions, typically 1/10th the size of the original.

Using the Napster client, a list of MP3 files on their hard drives is uploaded to a searchable database in real-time. In a quid pro quo situation, individuals agree to let others download music from their computers in return for letting them download music from others.

Napster itself stores no files. Only pointers to where those files may be found, indexed

using filenames, sizes and advertised download bandwidth.

For example, if someone wants to hear the latest “Cake” song, she simply logs onto the Napster server, does a quick search and is given a list of locations from where the song can be found. She selects the appropriate source and downloads the file directly from the remote PC to her hard drive. Voila, you can now listen to the song.

Naturally there are major copyright issues; major interests in the music industry are currently filing suits against Napster in the hope of shutting them down. However, these lawsuits are not about Peer-to-Peer technology per se. Rather they are about controlling content and who listens to that content and when.

This is an example of existing technologies being put to work in new ways. By leveraging the interconnected computing environment we know as the Internet, we have enabled a new paradigm, Peer-to-Peer computing. The concept is simple, allow individuals connecting to one another across the Internet, to perform useful tasks without the need for a central server to be involved in the actual transaction.

This is important. By bypassing the server from doing the actual work, you have now moved scalability from the server to the network.

Peer-To-Peer Computing

Napster is an example of P2P File Sharing. Another application using the same basic principles is P2P Computing; a form of distributed processing. P2PC works by breaking up complex tasks into smaller units and distributing those units among a pool of microcomputers. In this way, large and

complex problems can be solved using relatively meager amounts of processing power. P2PC builds metacomputers.

Unfortunately, not all problems can be mapped to this sort of architecture. Much of the research into distributed computing has concentrated on finding ways of making it a generic method of accelerating software rather than a specific way of writing programs. However, when the problem does map well, the performance gains can be remarkable.

Typically users install a software client that runs as a low priority background process. During periods when the host operating systems is not performing useful work, the client becomes active using that “idle” time to work on its’ piece of the overall problem.

What most people don’t realize is that their computers spend a great deal of time idly waiting for something to do. Fifty times a second, on average, your computer makes a decision about what to do next. Each of these decision points is known as a “timeslice”. How that time is allocated is a function of the operating system.

Tasks such as reading mail, creating presentations, and browsing the Internet do not require very much CPU. Most of the time is spent waiting around for the user to do something. In fact many operating systems, such as Windows NT and 2000, have a special process called “System Idle”. Its job is to send No Operation (NOP) commands to the processor when there is no work to be done, the equivalent of your brain sending “twiddle your thumbs” commands to your hands.

Well why on earth would you do that? Modern CPUs, such as Intel Pentium IIIs, AMD Athlons and Compaq Alphas, run at

hundreds to thousands of MHz per second, consuming as much as 75W of power when running at full capacity. CPUs themselves have no way of knowing when they should be idle, so they tend to run at full speed much of the time.

Operating systems are responsible for “scheduling” work for the CPU and are therefore in the best position to know when a CPU should be “idle”.

Think of NOP commands as a hint to the processor that it can turn parts of itself off, saving power, reducing heat, and potential increasing the working life of the silicon and it’s surrounding componentry. P2P takes advantage of these gaps making use of otherwise unused capacity to perform useful work.

How Idle Is Your Computer?

It is possible to view the idle task by running the Windows *TASKMGR.EXE* program found in your `\Windows\System32` directory (see figure 1). The system idle process can also be used to measure how inactive the computer is. For example in this case, the “System Idle Process” has spent 98% of its time doing nothing. Imagine if this idle time could be leveraged for something that could be beneficial to

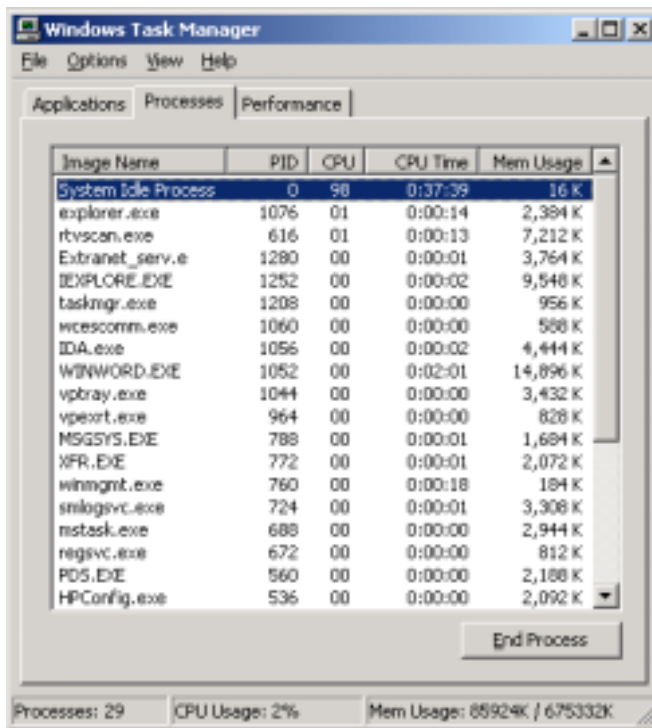


Image Name	PID	CPU	CPU Time	Mem Usage
System Idle Process	0	98	0:37:39	16 K
explorer.exe	1076	01	0:00:14	2,384 K
rfvscn.exe	616	01	0:00:13	7,212 K
Extranet_serv.e	1280	00	0:00:01	3,764 K
EXPLORE.EXE	1252	00	0:00:02	9,548 K
taskmgr.exe	1208	00	0:00:00	956 K
wcscomm.exe	1060	00	0:00:00	588 K
IDA.exe	1056	00	0:00:02	4,444 K
WINWORD.EXE	1052	00	0:02:01	14,896 K
vpray.exe	1044	00	0:00:00	3,432 K
vpsort.exe	964	00	0:00:00	828 K
MSGSYS.EXE	788	00	0:00:01	1,684 K
XFR.EXE	772	00	0:00:01	2,072 K
winmgmt.exe	760	00	0:00:18	184 K
smlogsvc.exe	724	00	0:00:01	3,308 K
mstask.exe	688	00	0:00:00	2,944 K
regsvc.exe	672	00	0:00:00	812 K
PDS.EXE	560	00	0:00:00	2,188 K
HPConfig.exe	536	00	0:00:00	2,092 K

large corporations with many PCs?

The basic principles behind P2P Computing are not new. P2P Computing could be viewed as the natural evolution of distributed computing in the Internet age.

Pioneers in Peer-to-Peer Computing

Several scientific teams and organizations, have been pioneering P2PC for several years

- The Great Internet Mersenne Prime-Search ([GIMPS](#)) – Started in January 1996. Their objective is to discover Mersenne prime numbers (2^n-1), of which there are only 38 known ones. To date they have discovered four, with the largest being $2^{6972593}-1$. This number contains 2,098,960 digits and is the largest prime number known.

In conjunction with GIMPS, the [Electronic Frontier Foundation](#) (EFF), is offering a prize of \$100,000 to the finder of a 10 million digit prime number. A prize of \$50,000 was awarded in April of 2000 to the GIMPS team after they found the first 1 million-digit prime number.

- [Distributed.Net](#) – Started in February 1997 to crack the RSA RC5-56bit key challenge, DN was successful in its efforts and managed to complete the competition by October of the very same year.

Currently DN is working on two different projects, an ongoing RC5 64bit key challenge, currently processing 141 gigakeys per second, and the Optimal Golomb Rules search; an obscure mathematical formula with implications for combinatorics, coding theory and communications.

Past projects have mostly involved cracking encryption algorithms, most notably DES and the CS-Cipher. One could say the key goal of the Distributed.net project has been to prove that weak encryption no longer acceptable, and projects to crack DES and weaker variants of RC5 have certainly influenced the US government in de-restricting the export of strong encryption technology.

- [PiHex](#) – started in March 1998 to calculate high orders of Pi by Colin Percival, a 19 year-old, 6th year math major at Simon Fraser University in Canada. Using P2PC, the team was able to calculate Pi to the Quadrillionth bit!

Peer-to-Peer Computing in the Spotlight

Perhaps one of the most well known examples of P2PC is the Seti@Home project (<http://setiathome.ssl.berkeley.edu>).

Seti@Home is an effort to support the Search for Extraterrestrial Intelligence (SETI) and their search for life on other worlds. Every day, SETI receives 35 gigabytes of radio signal information from the Arecibo radio telescope in Puerto Rico. Due to limited resources, SETI is unable to analyze the data at any great depth.

Seti@Home collects this data, where it is divided up into small chunks and distributed to computers all around the world for analysis.

In operation since April 1999, over 2 million computers have contributed their idle CPU time to this effort. Currently, Seti@Home's performance capacity exceeds 33 TeraFLOPS, which is almost three times faster than IBM's ASCI White supercomputer; which in 2000 was considered the worlds fastest supercomputer according to www.top500.org operating at a mere 12 TeraFLOPS!

The Coming P2PC Explosion

In the past 12 months, there has been an explosion of effort in the P2P space. Non-profit projects are still the most popular among users. However a significant number of for-profit organizations are beginning to

appear, bringing business models and a quest for dollars with them.

Some of the most notable projects include:

- [Folding@Home](#) - understand how proteins self-assemble (10/00)
- [Golem Project](#) – create artificial robotic life (9/00)
- [Casino-21](#) – develop global weather forecasts to predict global warming trends (12/00)
- [Popular Power](#) – Optimize the flu vaccine (1/00)
- [ProcessTree Network](#) – Analyzing gamma flux radiation (12/99)
- [Parabon](#) – Assist with cancer research (6/00)

Industry Attention and Standards

It is not enough that small corporations and companies develop tools to make Peer-to-Peer Computing a reality. The rate of change, while fast for the technology, would be very slow viewed from an industry point of view. This technology needs the attention of the industry giants in the field of computing to start looking at the technology and begin to develop standards for its use and implementation.

Intel generated this attention when it formed the [Peer-to-Peer working group](#) in August 2000. In the keynote speech kicking off this event, [Patrick Gelsinger](#), vice president and chief technology officer, Intel Architecture Group stated:

"Peer-to-peer computing could be as important to Internet's future as the Web browser was to its past. While the most visible impact of this model has been in consumer environments, peer-to-peer computing has the potential to play a major role in business computing as well. By adding peer-to-peer capabilities,



corporations can tap into existing TeraFLOPS of performance and terabytes of storage to make today's applications more efficient and enable entirely new applications in the future." 4

Clearly, Intel feels that this type of technology has the potential to have an impact on the world of computing. They do not speak without experience in this matter, as Intel has been using Peer-to-Peer computing technologies for the last 10 years in the process of building and verifying chip designs.

The goal of the Peer-to-Peer Working Group is "to develop infrastructure standards to enable peer-to-peer computing everywhere." Their charter to accomplish this is "to determine areas for standardization, rapidly develop specifications, and promote adoption of these specifications as standards throughout the computer industry."

The working group is currently made up of 18 various corporations including HP. More details of this organization can be found at <http://www.peer-to-peerwg.org>.

The Social Benefits of P2P

One of the social aspects that is perhaps overlooked about P2P computing is that it allows a sense of continuity, community and contribution. Much like the ad campaigns of World War II, people can derive a feeling of satisfaction from contributing their resources to a goal which is greater than the sum of the parts. For example, a poll by Entropia, the people who run the GIMPS effort, discovered that as many as 26% of the participants in the GIMPS project felt that it was a good use of CPU time. A potential monetary reward was a secondary benefit; only 2% in the poll listed this as their primary concern.

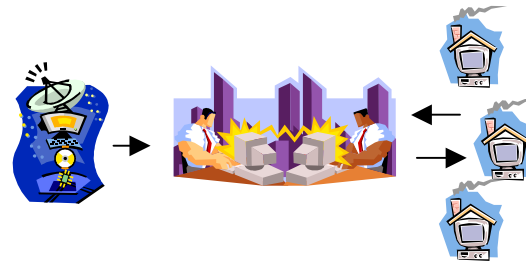
P2PC Application Models

So far, two broad application models have emerged for making use of P2P Computing. The first involves emulating traditional super computer architecture in the wide area. The second takes advantage of the characteristics of being distributed.

United We Stand, Together We Process

SETI@Home is a perfect example of this environment. Data is collected at Arecibo and then sent via magnetic tape to Berkley.

Information is broken into manageable chunks of 256KB each. Clients contact the server and request chunks, running advanced fast Fourier analysis algorithms whenever the computer begins running it's screensaver.



When the client is finished it contacts the server again, uploading its' results and downloads a fresh batch so the work can continue. The central servers aggregate the results and forward any interesting finds to the scientists involved.

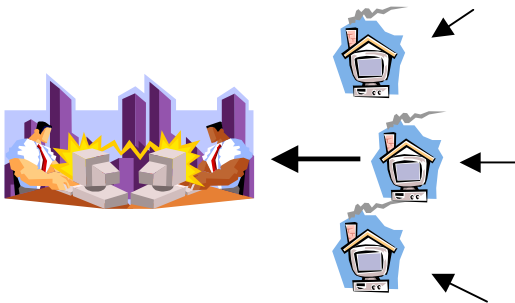
Centralized models appear best suited to applications that require the same set of operations to be repeated time and time again to different parts of the dataset. Tasks such as data mining, finite analysis, rendering computer graphics, and statistical analysis fall into this category. It best

emulates a massively parallel computing device, where each node is only in communication with a coordination processor part of the time.

However one of the disadvantages of this architecture is that has a fairly significant amount of administrative overhead. Splitting up data into chunks, sending and receiving that data over the network and performing the final assembly, requires considerable effort and coordination. This leads to subtle problems creeping up. For example when should an errant block of data be declared as missing, so it may be added back to the pool? It is this sort of issue that makes a distributed computation challenging.

Network Power

In the second model designers are taking advantage of the remote and usually widely dispersed nature of many P2PC networks, by turning a systemic characteristic into a veritable feature.



In most P2PC designs, each node is a processing unit of a much larger conglomerate. However, in addition to being a source of processing power each node is also a member of the global Internet.

It is possible to leverage this characteristic and turn a station into a bellwether, using it to measure characteristics about the network which when combined give us a global

picture of whatever conditions we may be interested in.

Once again each station in the network runs a special client program. However instead of downloading units of data to process, each client contacts a central server infrastructure, which gives it a list of measurements to be collected that day.

These tasks may include polling a particular set of web servers on a regular basis, or measuring the delay between the host itself and some prescribed location on the network. Once this data is collected it is sent at regular intervals back to the central server for aggregation into a report.

[Porivo](#) is a good example of this sort of application. Their product, [peerReview](#), provides a distributed web performance testing service. Customers purchase the service and identify the web sites they are interested in measuring. Periodically, Porivo's network of computers, connect to the required web sites measuring a number of predetermined metrics over a particular time period. Once the data is collected it is returned to the server for aggregation and final analysis and reported back to the customer.

For this sort of architecture to work, every node in the network must be able to measure the same set of characteristics. For example, applications that require a specialized piece of measuring equipment would severely limit the use of this product.

However one could imagine interesting applications such as those related to temperature measurement, air quality, light intensity, rainfall, and even noise levels all indexed by zip code if all participants could procure a measurement device cheaply enough.

Also, since these efforts are usually for profit, incentives must be provided when simple altruism is not the driver. In Porivo's case this involves the use of cash prizes. Every month random Porivo node providers win expensive items or cash prizes for providing CPU time to the network. Sometimes this can involve as much as \$10,000; not bad for simply leaving your computer on for a month.

Gremlins, Saboteurs and Crackers

One of the biggest problems with P2P Computing in open environments has been bugs and malicious data entry. For example, bugs in the client could introduce problems into the data set corrupting results and invalidating the exercise.

Scrupulous testing is required to ensure that everything is functioning correctly to prevent poisoning the work. [Seti@Home](#) was recently quoted in the New York Times that during some months in its past, up to 50% of its resources were dedicated to finding out people who were cheating the system. "As soon as you offer any kind of incentive, you will invite cheating"⁵ said Armin Lenz in that same article, a former executive at a commercial distribution computing company.

Another problem, which has affected some of the efforts of Distributed.net for example, have involved malicious data submissions. DN has a scoreboard, and a team system. Individual's are ranked according to how many blocks they've completed. Individuals can also form teams and these are ranked in a separate score board.

Last year, DN discovered a team from [Russia](#) was submitting more results than they could realistically have calculated.

Investigations discovered they were using a hacked client reporting fake results. The subterfuge was quickly caught, but this could have ruined the entire experiment.

P2P businesses need to be mindful of this problem and ensure that they have checks and balances to prevent bad or false data being submitted to the system. This would be an excellent form of corporate espionage for example, ruining results and perhaps delaying projects for critical days, weeks or years from completion due to false data.

Other malicious examples of P2PC usage are the infamous back orifice and Trojan horse programs such as trinoo, stracheldraht and TFN. Crackers break into various computers on the net and install special clients.

These clients wait for instructions from the attacker using special encrypted messages. When the cracker has collected enough machines they can securely launch attacks in a distributed manner against others using untraceable network resources in order to distract pursuers or provide cover while they go after their real target. Ironically this is also a form of P2PC computing, albeit one with malicious intent.

Peer-to-Peer Computing Business Models

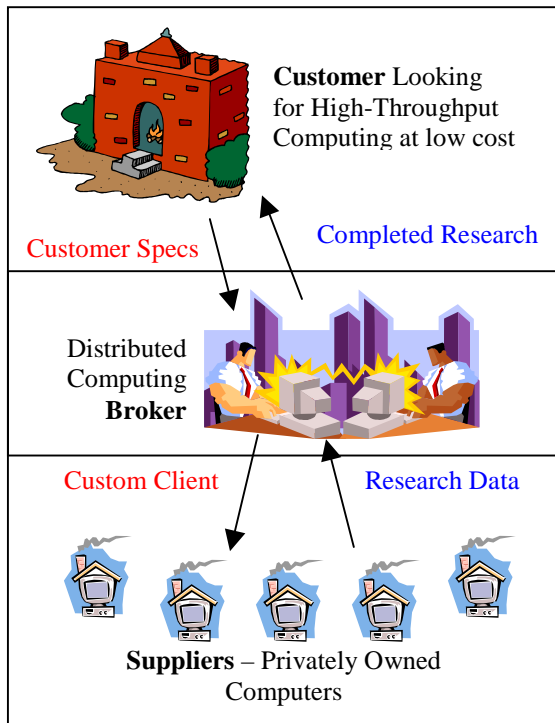
While the above non-profit sites are primarily focused on the research needs of the scientific and mathematic communities, (along with philanthropic desires of Internet computer users), there are several new organizations that are experimenting with the development of revenue generation.

So far two ways of making money out of P2P are emerging. The first involves being paid for organizing and "farming" the resources. The second is a pure

infrastructure play, providing the tools to build the required capacity. This is similar to provisioning networks, some make money being the ISP, others create revenue by creating the tools and equipment necessary to be an ISP.

Model #1: Brokering

The first model focuses on positioning a P2P company as middleman, or “broker” of distributed computing resources. *Brokers* have to develop technology frameworks that allow them to easily distribute compute intensive customer problems.



Rather than attempting to solve a specific problem, their technology has to provide a virtual machine environment, which can support a wide variety of problems on a wide variety of platforms.

Brokers advertise the availability of supercomputing capabilities for a fraction of the cost of a conventional supercomputing

solution, targeting companies with short-term needs for the massive processing bandwidth they can yield from their network of *Suppliers*.

Customers contract with *Brokers* to write or port an application specific to their needs. Depending on the fee the task is distributed out to as many nodes as required. For billing reasons figuring out the acceptable “success” or “failure” threshold is critical.

Once the contractual elements are figured out, the *broker* would then develop a custom solution using an appropriate application model. Nodes would then be contacted via email and asked to download the new client application or in an ideal situation automatically check for new downloads at regular intervals. Once a new task had been detected, downloading and running would be automatic.

Challenges: Auditing and Accounting

Brokers are responsible for keeping track of how much work each *supplier* has contributed; any hint of deceit destroys the whole basis for the system. Based upon this value, a fair level of remuneration can be made to each *supplier* on a regular basis. The trick is to pay enough out to the *suppliers* and charge enough from the customers that a) *customers* choose them over a custom solution and b) suppliers will willingly turn over unused cycles to them leaving their computers running day and night.

Safe guards to prevent fraud at all levels must be included in the software. The *Supplier* should receive accurate and timely information about how much processor time has been supplied at all times with a pre-agreed set of rates and tariffs or incentives like prizes and cash payouts. The *Broker*

needs to ensure that clients have not been tampered with; in order to increase pay out, and that the results are fair and true.

Brokers can make money on both ends, since they play both the Paymaster and the Engineer, providing unique solutions for the each *Customer*. *Suppliers* make money for leaving their computers on, and *Customers* save money from not having to buy and operate a Super Computing facility.

Customer Billing

Customer Billing is probably one of the most fascinating elements of this model. For example, should *Customers* be billed for absolute CPU time utilized, or the number of nodes involved, or in a tiered fashion with the greatest sums going to those who return the required results the quickest? Depending on the billing model, different auditing rules will be required to ensure that *Brokers* are not cheating *Customers* by falsely inflating numbers.

A good business example of the Broker model is [Distributed Science](#). Their web home page states their charter to be:

“If you need a peer-to-peer application that spans your corporate intranet, or even the Internet, we should be talking”.¹

Companies contracting with Distributed Science can either choose to implement a custom application within their own network of PCs, or allow Distributed Science to perform the actual number crunching with the 137,000 or so Internet connected PCs already signed up to be part of the [ProcessTree Network](#).

Paying Suppliers

On November 17, 2000, the ProcessTree Network announced that they had signed up

their first *Customer*. All those *Suppliers* who had previously registered with ProcessTree [received an email](#) stating that it was ready to start working for hire, and that *Suppliers* that met certain needs would be eligible for payment for services rendered.

“The good news is that as soon as the end of the month, some of the suppliers of ProcessTree will be able to earn money with their computers...”

The job is a quality-of-service monitoring system that allows real-time checking on the performance and availability of websites. We will initially start with 25 locations, for each of which we plan to have a number of suppliers to provide a 24/7 coverage from each location. Each such group in a location we call an “ideal machine”.²

Like ProcessTree, each company involved in this business model expects to provide micro-payments or credits to a *supplier's* account for each segment of *work* that their computer completes.

While each segment itself will not result in a large cash payout, over time, the *brokers* expect that the cash payout could be used to cut a *Supplier's* ISP bill on a regular basis of around \$10-\$20 per month. Others intend to provide incentives such as the chance to win prizes based on exactly how much effort their computer has contributed.

Model #2: Build your own with our tools

While some companies would feel comfortable making a “buy” decision and utilizing the services of brokers, others feel that they have a large enough need for a super computer style facility that building their own would be a better investment.

This is yet another niche for P2P companies to inhabit. By providing the tools to build,



test and distribute this style of application companies can save millions of R&D dollars by getting their product to Market sooner.

[Applied Meta](#) (AM) is one such company. Originally a research project at the University of Virginia for the Department of Defense and Department of Energy, AM's product Legion creates a distributed operating system environment.

Legion allows developers to use a variety of languages, and underlying operating systems, abstracting away the details of the actual implementation into the space of a traditional operating environment with schedulers, memory spaces and communications.

The commercial version allows computers as diverse as Intel PC up to esoteric supercomputers to be linked into a single environment. As stated on their web site about the Legion product:

"LEGION is Applied Meta's enterprise solution that enables all networked resources to **Work as One**. No matter what the platform, language, or location, LEGION enables networked resources (hardware, software, applications, data, and people) to Work as One.

LEGION provides an enterprise level environment that allows working relationships to exist between trillions of heterogeneous objects."³

Current installations of Legion include many universities such UC Berkley, UC San Diego, the NSCA in Illinois, several NASA research facilities and the Department of Defense.

Much more detailed information is available about this environment at [Applied Meta's Whitepaper website](#).

[Porivo](#), has also developed similar technology called "[peerPlane](#)", which is it marketing as a full Peer-to-Peer Computing development platform supporting both centralized and distributed data models. At this time Porivo is targeting enterprise customers while Applied Meta works with the scientific community.

Traditional Development Funding Model

In the case of Legion, and others providing the same type of service, funding comes from the sale and support of the development environment. The organization will work with its customers to help identify the business needs, and the best way to write application code to solve those needs. In all cases, the organization selling the development environment is willing to sell the buyer consulting time to help them best take advantage of the new development environment.

What areas could P2P Computing impact?

There are several other examples of where Peer-to-Peer Computing could have an immediate impact.

- **Network load monitoring** data collection process for HP's OpenView product. Instead of requiring the large OpenView servers to be monitoring and collecting network information, let the computers that reside on each subnet do the data collection. On regular intervals, they would send that information to the main OpenView analysis servers.

This would utilize the Peer-to-Peer Computing distributed data model, which is more efficient. As each computer would do its own data collection it would greatly reducing the

complexity of developing such an application.

- **Software testing and regression analysis** could be implemented quickly. Instead of creating a lab with many client computers, send out the software to be tested to actual PCs within a company's infrastructure. It would allow for the testing of the software on any platform required, as all of them would be available somewhere within the company.
- **Application sharing to small form-factor devices.** Think of this as Peer-to-Peer ThinClient. Devices that do not have access to full-powered Office applications could take advantage of the processing capabilities of nearby PCs.

Through a wireless connection, one could utilize ThinClient concepts to access full-powered applications via the PCs "logically" nearby the PDA. You could take your Bluetooth-enabled Jornada PocketPC to a conference room and have full access to PowerPoint for your presentations.

Your PDA would act as the "mediator" between a desktop PC sourcing the application, and the Bluetooth-enabled display providing the visual content (this technology, while not available today, is approximately only 1 year away with working demos now in existence).

- **Multimedia Post-Production data Management** to increase virtual bandwidth of data streaming or the encoding of and processing of video in and effects in real time.
- **Personal Data Agent** to monitor local computer activity for keywords, and to

provide insight and enhanced data points regarding those keywords

- Real-time visibility to **Financial Data** by running the "month-end" close each night using a network of P2PC-enabled PCs.
- **Decentralized Web Hosting** – instead of a single web server, you could link multiple systems together and provide the hosting of information from those systems. Requests for access would go to the least busy system, or possibly the closest system to the party requesting access.

What Security Issues Exist with P2P Computing?

If businesses were to involve itself in Peer-to-Peer Computing, the question of security would have to be paramount. All of the serious vendors of Peer-to-Peer Computing identify security as an issue they are both concerned with and are dealing with.

For instance, Porivo's PeerPlane technology has a separate module called PEER Security Manager. It is designed to protect the user's data and network access by regulating the communication between the Java APIs on the user's PC and the project code being run on the server. Security policies can be user defined based on each project's needs, and all communication between the peer and the server can be encrypted using SSL.

Applied Meta's Legion product has focused on security from its inception and utilizes strong authentication using public key technology and signed certificates, security policies independent from the computing mechanism allowing rapid deployment, group based security policies, trust relationships, etc.



References

1. Distributed Science Homepage (<http://www.distributedscience.com/>), 5th paragraph
2. ProcessTree Homepage (<http://www.processtree.com>), “News and Projects” link, November 21, 2000, 3rd and 4th paragraph.
3. Applied Meta Homepage (<http://www.appliedmeta.com>), “About Legion” link, 1st and 2nd paragraph.
4. Intel Press Release Archive Homepage, (<http://www.intel.com/pressroom/archive>), “Intel Forms Peer-To-Peer Working Group”, August 24, 2000, 2nd paragraph.
5. [The Search for E.T. Yields Earthly Cheats](#), New York Times, May 24, 2001, J.D. Biersdorfer

WWW Pages For Companies and Organizations Mentioned

1. Napster – <http://www.napster.com/>
2. GIMPS - <http://www.mersenne.org/prime.htm>
3. PiHex - <http://www.climate-dynamics.rl.ac.uk/home.html>
4. Distributed.Net - <http://www.distributed.net>
5. Seti@Home - <http://setiathome.ssl.berkeley.edu>
6. Folding@Home - <http://www.stanford.edu/group/pandegroup/Cosm>
7. Golem Project - <http://www.demo.cs.brandies.edu/golem>
8. Casino-21 - <http://www.climate-dynamics.rl.ac.uk/home.html>
9. Popular Power - <http://www.popularpower.com>
10. ProcessTreeNetwork - <http://209.240.46.125>
11. Parabon - <http://www.parabon.com/>
12. Porivo - <http://www.porivo.com>
13. Distributed Science - <http://www.distributedscience.com>
14. Applied Meta – <http://www.appliedmeta.com>
15. Intel – <http://www.intel.com>
16. Intel Peer-To-Peer Working Group – <http://www.peer-to-peerwg.org>
17. PC COE Client Inventory Database - <https://pawnt092.corp.hp.com/CIBWarehouse/>
18. Hewlett-Packard Unix Server Homepage - <http://www.unixservers.hp.com>
19. Standard Performance Evaluation Corporation – <http://www.spec.org>
20. Platform Computing – <http://www.platform.com>
21. O’Reilly Conference on Peer-to-Peer - <http://conferences.oreilly.com/p2p>
22. DCI’s Summit on Peer-To-Peer Computing - <http://www.dci.com/events/p2p>

Additional WWW Pages for Additional Organizations involved in P2P Computing

1. United Devices - <http://www.uniteddevices.com>
2. GRISK - <http://www.ii.uib.no/grisk>
3. Condor - <http://cs.wisc.edu/condor>



Appendix A – Processor Performance Comparisons

Benchmarking Processors

It is possible to roughly compare the raw capabilities of processors against the performance of other processors. Benchmark data on a wide range of systems are published on sites such as the [Standard Performance Evaluation Corporation](#) (or SPEC). SPEC has published a series of benchmarks that test and document the compute intensive capabilities in both integer ([SPECint95](#)) and floating point ([SPECfp95](#)) math for all types of CPUs and compute systems. In addition, the more recent benchmarking tool SPEC2000CPU is being used as it can handle the most current hardware and capabilities. Computing manufacturers have performed measurements of their systems and have [published them on the SPEC website](#).

For the purposes of this comparison, the floating point benchmarks are used since they reflect the strong suit of high end systems typically used for supercomputing devices, and they best represent the type of computing that would be required in a Peer-to-Peer Computing model.

The SPECfp95 value identifies the relative speed of the CPU when compared to a baseline system performing the same test. Higher numbers indicate faster performance. In addition, servers tend to scale almost linearly with multiple CPUs. Thus going from 1 CPU to 2 will double eSPECfp95 performance results.

Processor	int2000	fp2000	int95	fp95
166Mhz Pentium/MMX			5.6	4.34
266Mhz Pentium Pro			7.73	6.57
433Mhz Celeron			14.9	10.9
450Mhz Pentium II			18.5	13.3
500Mhz Pentium II Xeon			21.6	15.9
650Mhz PIII	299	215		
1Ghz PIII	442	284		
1.7Ghz Pentium 4	586	608		
500Mhz SGI Origin3200 R14K	427	463		
750Mhz HP J6700 PA-8700	603	581		
900Mhz Sun UltraSPARC-III	467	482		

Table 1: SPEC Performance Info for Various Processors

Peer-to-Peer Computing Equivalency Formula

Using these benchmark numbers it is possible to roughly determine a formula that compares the number of processors required to equal the performance of another type of processor, assuming the idle cycles are being harvested from the first type of processor.

$$\text{Count}^{S1} = \frac{1}{\text{perf}^{S1} / \text{perf}^{S2} * \%^{AV} * \%^{RC}}$$

where

- Count^{S1} = number of “S1” systems required to equal the performance of a single “S2” system.
- fp^{S1} = performance of S1 system
- fp^{S2} = performance of S2 system
- $\%^{AV}$ = percent of idle CPU time available on S1
- $\%^{RC}$ = percent of idle CPU time able to be harvested from S1

$\%^{AV}$ can be described as the average percent of time available on any system for use within an P2P Computing environment on tasks other than those associated with the local user. Measurements indicate that this number could be as high as 90% available.

$\%^{RC}$ can be described as the percent of time that can be dedicated toward P2P computing tasks on a source processor. It accounts for the fact that a percent of the percent idle time is used up on overhead tasks associated with a P2PC project such as data transmission, system overhead, remote communication, etc.

Example of Comparison between Processors

Assume your company had 500 PCs containing 450Mhz Pentium II processors, and another 500 PCs containing 650Mhz Pentium IIIs. Measurements had been performed that indicated, on average, their CPUs were 90% idle. The software you were using for P2P Computing also required a 30% overhead value. You were considering purchasing several large Pentium 4 servers, but wanted to review how much spare CPU time you could harvest from your installed base of computers before you went out and made those purchases.

Assumptions:

- S1 = 500 450Mhz Pentium II
- S1 = 500 650Mhz Pentium III
- S2 = 1.7Ghz Pentium 4
- %^{AV} = 90%
- %^{RC} = 70% (100% of available time – 30% overhead)

Processor	int2000	fp2000	int95	fp95
450Mhz Pentium II			18.5	13.3
650Mhz PIII	299	215		
1.7Ghz Pentium 4	586	608		

Table 2: P2P Computing Processing Equivalencies

Pentium II Intermediate Measurements

$$\frac{1}{18.5 / 586 * .9 * .7} = 50$$

Pentium III Intermediate Measurements

$$\frac{1}{299 / 586 * .9 * .7} = 3$$

Final Results

Using the values determined above, the performance available in a 1.7Ghz Pentium 4 is equaled by the performance associated with 50 450Mhz PII, or 3 650Mhz PIIIs. Therefore:

500 450Mhz Pentium IIs = 10 1.7Ghz Pentium 4's
 500 650Mhz Pentium IIIs = 166 1.7Ghz Pentium 4's

The processing power available to this business within its installed base of **1000** computers is equal to **176** 1.7Ghz Pentium 4's.