Talks

◆ P3: Personal Power Plant
  - Makes over your PCs into power generator on the Grid
    ■ 20 min

◆ Integration patterns of P2P and Grid Technologies
  ■ - 10 min

By Kazuyuki Shudo <shudo@ni.aist.go.jp>
■ GTRC (Grid Technology Research Center),
  AIST (National Institute of Advanced Industrial Science and Technology)
P3: Personal Power Plant
Makes over your PCs into power generator on the Grid

Kazuyuki Shudo <shudo@ni.aist.go.jp>,
Yoshio Tanaka,
Satoshi Sekiguchi
Grid Technology Research Center, AIST, Japan
P3: Personal Power Plant

Middleware for distributed computation utilizing JXTA.

Traditional goals
- Cycle scavenging
  - Harvest compute power of existing PCs in an organization.
- Internet-wide distributed computing
  - E.g. distributed.net, SETI@home

Challenging goals
- Aggregate PCs and expose them as an integrated Grid resource.
  - Integrate P3 with Grid middleware? cf. Community Scheduler Framework
- Dealings and circulation of computational resources
  - Transfer individual resources (C2C, C2B) and also aggregated resources (B2B).
  - Other resources than processing power.
  - Commercial dealings need a market and a system supporting it.
P2P way of interaction between PCs

- **P3 uses JXTA** for all communications
  - JXTA is a widely accepted P2P protocol, project and library, that provides common functions P2P software requires.

- **P2P concepts supported by JXTA efficiently support P3:**
  - **Discovery**
    - PCs dynamically discover each other and jobs without a central server.
  - **Peer Group**
    - PCs are grouped into job groups, in which PCs carry out code distribution, job control, and group communication for parallel computation.
  - **Overlay Network**
    - Peer ID in JXTA is independent from physical IDs like IP addresses and MAC addresses.
    - JXTA enables end-to-end bidirectional communication over NA(P)T and firewall (even if the FW allows only unidirectional HTTP).
    - This function supports parallel processing in the message-passing model, not only master-worker model.
Other Benefits from JXTA

- **Scalability**
  - JXTA Project set its scalability target as 300,000 peers are active in 1,500,000 peers.

- **Configuration-less**
  - A P3 peer can discover other peers and submitted jobs with JXTA’s discovery function.

- **Multi-protocol**
  - JXTA relay peers mediate messages between TCP, HTTP, IP multicast and possibly other protocols like Bluetooth.
Design Goals

- **Application neutral**
  - cf. Client software of traditional dist. comp. projects (e.g. distributed.net) is tightly coupled with a few applications.
  - P3 is decoupled from applications and users can submit apps into a PC pool.

- **Practical**
  - not only for research.
    - There have been many many middleware for research purpose.
    - Development of P3 is funded by a government agency to promote the development of economy.
  - A Protein-Folding application is working on P3 and a signal processing app is planned. Practicality of P3 has been improved with them.

- **Scalable**
  - We could test P3 with only dozens of PCs so far.
  - But we’re measuring other scalability factors including throughput of workunit-processing by a master.
Design Goals (cont’d)

NA(P)T and firewall traversable

- Now, Most PCs are located behind a firewall on the Internet.
- To overcome this restriction, many dist. comp. systems use only HTTP as communication protocol and limit communications to one-way (client -> server).
  - e.g. United Devices' GridMP

- P3 uses JXTA for all communications
  - All P3 peers can communicate with each other bidirectionally.
  - P3 provides a message-passing API besides master-worker API.
Design Goals (cont’d)

Choice of applications by PC providers
- PC providers (participants in a dist. comp. project) should be able to choose jobs to which their PCs are devoted.
  - It is very important for PC providers to be able to control their own resources.
- In a traditional Internet-wide project, a PC provider has only one choice, install or not.
- Using P3, a PC provider can confirm a digital signature of a job and decide whether to accept it or not.

Adaptation to both intra- and Internet
- On the Internet, we have to assume that there are malicious PC providers.
  - they will try to cheat the software and the operators of the project. E.g. pretending to finish calculation, DoS attack and so on.
- P3 can confirm the correctness of collected results by voting.
  - Distribute identical workunits and verify the returned results.
  - This function can be disabled and a verifying logic can be substituted.
Design Goals (cont’d)

Easy deployment and automatic updating

- The amount of installation and updating labor are proportional to the number of PCs and can be huge.
- Vulnerable client software will be mostly left as it is if the software cannot be updated automatically somehow.
  - A vulnerability was found in SETI@home client software in April 2003.
- P3 can be installed by only mouse-clicks on a web page and updated automatically.
  - cf. Java Web Start (JWS)
Structure of P3

- **P3** currently consists of three subsystems:
  - Job management, Parallel programming library, Job monitor

- **Job management subsystem**
  - Host jobs (submitted apps) and control their execution.
  - *Host*: A daemon program runs on a provided PC.
  - *Controller*: by which a resource user submit and control jobs.

- **Parallel programming libraries**
  - Application programs that use these libraries can run on P3.
    - Master-worker
    - Message Passing (like MPI)

- **Job monitor**
  - Shows the state of jobs and Hosts
  - Web-based
Job Management Subsystem: Controller

- A resource user submits and controls jobs with Controller.

Image: A screenshot of the P3 Controller interface showing "Attending Hosts" and a "submitted job" entry for "Protein Folding (MMG)."
Job Management Subsystem: Host

- A daemon program runs on a provided PC.
  - A Host can be invoked in a head(GUI)-less mode. In that case, it decides whether to join a found job or not according to a policy supplied by the PC provider (owner).
  - Host can host multiple jobs simultaneously.

Discovered jobs
- Output from a running job
Job Management Subsystem: Job Monitor

Web browser

Total view

Host view

Number of processed workunits

Calculation speed

Table:

<table>
<thead>
<tr>
<th>Job Name</th>
<th>Start Time</th>
<th>End Time</th>
<th>CPU Time</th>
<th>Average CPU Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job1</td>
<td>03/04/01</td>
<td>03/04/01</td>
<td>0.54</td>
<td>0.54</td>
</tr>
<tr>
<td>Job2</td>
<td>03/04/01</td>
<td>03/04/01</td>
<td>0.55</td>
<td>0.55</td>
</tr>
<tr>
<td>Job3</td>
<td>03/04/01</td>
<td>03/04/01</td>
<td>0.56</td>
<td>0.56</td>
</tr>
<tr>
<td>Job4</td>
<td>03/04/01</td>
<td>03/04/01</td>
<td>0.57</td>
<td>0.57</td>
</tr>
<tr>
<td>Job5</td>
<td>03/04/01</td>
<td>03/04/01</td>
<td>0.58</td>
<td>0.58</td>
</tr>
</tbody>
</table>

Graphs showing calculation speed and other performance metrics.
Job Management Subsystem:
Job Monitor (cont’d)

Job Information

Host Information

<table>
<thead>
<tr>
<th>Rank</th>
<th>name</th>
<th>Result Received [WU]</th>
<th>CPU Time</th>
<th>Average Calc Speed [sec/WU]</th>
<th>Fastest Time per WU [sec]</th>
<th>Average Calc Time [sec/WU]</th>
<th>OS</th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>sumomo11 - shudo</td>
<td>2711</td>
<td>0:54:29</td>
<td>1.24</td>
<td>1.09</td>
<td>1.21</td>
<td></td>
<td>i386</td>
</tr>
<tr>
<td>2</td>
<td>sumomo12 - shudo</td>
<td>2653</td>
<td>0:50:35</td>
<td>1.27</td>
<td>0.82</td>
<td>1.14</td>
<td></td>
<td>i386</td>
</tr>
<tr>
<td>3</td>
<td>sumomo14 - shudo</td>
<td>2523</td>
<td>0:36:18</td>
<td>1.33</td>
<td>0.76</td>
<td>0.86</td>
<td></td>
<td>i386</td>
</tr>
<tr>
<td>4</td>
<td>sumomo13 - shudo</td>
<td>2172</td>
<td>0:40:33</td>
<td>1.55</td>
<td>0.77</td>
<td>1.12</td>
<td></td>
<td>i386</td>
</tr>
<tr>
<td>5</td>
<td>sumomo15 - shudo</td>
<td>1916</td>
<td>0:29:21</td>
<td>1.76</td>
<td>0.77</td>
<td>0.92</td>
<td></td>
<td>i386</td>
</tr>
<tr>
<td>6</td>
<td>sumomo07 - shudo</td>
<td>1262</td>
<td>0:23:42</td>
<td>2.67</td>
<td>0.83</td>
<td>1.13</td>
<td></td>
<td>i386</td>
</tr>
<tr>
<td>7</td>
<td>sumomo06 - shudo</td>
<td>837</td>
<td>0:12:57</td>
<td>4.02</td>
<td>0.82</td>
<td>0.93</td>
<td></td>
<td>i386</td>
</tr>
<tr>
<td>8</td>
<td>sumomo05 - shudo</td>
<td>419</td>
<td>0:06:25</td>
<td>8.03</td>
<td>0.82</td>
<td>0.92</td>
<td></td>
<td>i386</td>
</tr>
</tbody>
</table>
How Peer Group is utilized

- **Net Peer Group**
  - A PG always exists in a JXTA apps.
- **Base Peer Group**
  - A PG for P3.
  - All Hosts and Controllers join this PG first.
- **Job Peer Group**
  - A PG for each job.
  - All job-related comm. are performed in this PG.
    - Job control
    - Parallel processing

Net Peer Group
(always existing JXTA’s base group)
Job Submission by Controller

(1) Create a Job Peer Group

(2) Join the Job Peer Group

(3) Share application code in the group with JXTA CMS service
Participation in a Job

1. Discover Job Peer Groups
2. Decide to join a discovered job
3. Join the Job Peer Group
4. Discover Application code
5. Obtain the code from a Controller
Parallel Programming Libraries

- Application programmers can use 2 libraries:
  - Master-worker
  - Message passing (like MPI) - JXTA-MPI

- Emulator
  - enables us to run parallel apps on one PC.
  - It is extremely useful to test and debug the application in advance of real deployment.

Diagram:
- Application
  - Master-Worker API
    - Master-Worker Library
  - Message Passing API
    - Message Passing Library
- Other Libs
- Object Passing Emulator
- Object Passing Library
- P2P comm. Library: JXTA
Performance Evaluation

- JXTA provides a rich set of functions, but... Isn’t it slow?
  - Certainly, not fast. But enough for many cases.
- Performance measurements:
  - Basic communication performance
    - Latency and throughput
  - Application
    - RC5 attack
- Environments:
  - 2.4 GHz Xeon PCs, Gigabit Ethernet
  - Linux 2.4.19, Java 2 SDK 1.4.2, JXTA 2.1
  - Rich PC and network compared with today’s Internet, but in which limits of P3 software can be measured clearly.
Communication Latency

1 byte round-trip communication. A one-way comm. takes

- TCP (in C): 0.062 msec
- TCP (in Java): 0.064 msec
- P3’s Message passing: 4.5 msec

Not fast

- It can limit the number of workunits that a master can process. One workunit takes several milliseconds.
- Enough for many situations, but JXTA should be improved.
Communication Throughput

- Message passing library is used.
- About 100 Mbps (100 x 10 ** 6 bps).
  - Not very fast on Gigabit Ethernet, but P3 can fill Internet connections to small offices and homes.
- Throughput declines with larger messages.
  - Such a large message should be divided.

![Graph showing throughput vs. data size]
Application Performance

A load test with small workunits.
- Brute-force attack on RC5 cryptsystem.
  same as distributed.net working on RSA RC5 challenge.
- P3 is tolerant of such granularity of workunits (taking several seconds)
  with dozens of PCs.

Granularity of workunit
- \(0x8000\): 1.4 sec
- \(0x4000\): 0.69 sec
- \(0x2000\): 0.36 sec

Very fine for a load test. Unusual for Internet-wide computation.
Related Work

- **JNGI**
  - being developed by Sun Microsystems.
  - uses JXTA.
  - utilizes peer groups to manage many PCs efficiently.
    - Groups of working peers are established in advance, and a submitted (parallel) job is assigned one of those groups.
    - cf. while P3 creates peer groups for each job.
  - Though a paper has been published (in GRID 2002), most part of the idea has not been implemented (as of Sep 2003).

- **XtremWeb, GreenTea, Javelin, Bayanihan, ...**
  - Of course, they have their own good points, but
  - PC providers cannot choose application programs.
  - Programming model is limited to master-worker or divide-and-conquer.
  - Firewall are not considered.
    - use Java RMI, TCP and so on.
  - Not tolerant of malicious PC providers or obscure.
Future Plan

- **Public release**
  - 2Q 2004 planned

- **Test with more PCs**
  - on the testbed of a regional Grid partnership?
    - Over 1,000 procs at AIST, TITECH, Waseda U. and other organizations.
  - How do we construct super-peer network, tree of RendezVous peers of JXTA?
    - It's not certain for JXTA to work with such number of peers.

- **More applications and deployment**

- **Write a paper**
  - There are currently these slides, SC2003 poster and papers in Japanese.
Integration Patterns of P2P and Grid Technologies

Kazuyuki Shudo <shudo@ni.aist.go.jp>,
Grid Technology Research Center,
AIST, Japan
Synergies between Grid and P2P

- It seems to be an intuitive feel.
- There have been activities including:
  - General and conceptual discussion
  - Introduction of activities: projects and implementations
  - Key capabilities of these technologies
  - Comparison in major aspects

- GGF9 workshop (Oct 2003)
  - Peer-to-Peer and Grids: Synergies and Opportunities
  - http://www-csag.ucsd.edu/P2P-Grid/

- APAN and Internet2 joint meeting (Jan 2004)
  - P2P and GRID: Convergence and Challenges
  - http://apan.net/meetings/honolulu2004/ws-application.htm#appl3

- This talk concentrates in instances of Grid/P2P integration
Grid/P2P Technologies and their Integration

Technologies and characteristics regarded as belonging to P2P
- Resource discovery
  - Ad-hoc and dynamic grouping/matching
- Overlay network
  - Logical ID, NA(P)T traversal, multi-protocol support
- Scalability

Grid
- Resource aggregation
  - Cluster of computers, global filesystem, ...
- Scheduling
- Parallel/distributed processing
- (PKI-based) authentication/authorization
Preliminary Classification toward Taxonomy

**Instances**

- **JNGI, P3**
  - Distributed computing system based on JXTA
- **Sun ONE Grid Engine (SGE) + JXTA**
  - SGE is a job management software for PC/WS cluster like PBS and Platform’s LSF
- **P2P Grid system + Grid middleware**
    - Platform’s LSF + Globus Toolkit 3.x (GT3)
    - United Devices’ Grid MP was planned to be compliant with the CSF

**Classes**

- **Grid over P2P**
- **P2P over Grid**
- **Cooperation**
Grid over P2P pattern

- JNGI and P3

Grid Technologies

- Resource aggregation
- Parallel processing

P2P Technologies

- Resource discovery
- Overlay network

- Aggregate resources with P2P technologies
P2P over Grid pattern

- Sun ONE Grid Engine (SGE) + JXTA

**Grid Technologies**

- Resource discovery
- Overlay network
- Resource aggregation
- Parallel processing

**P2P Technologies**

- Resource discovery with JXTA
- JXTA team and SGE team started this collaborative work.
Cooperation pattern

P2P Grid system + Grid middleware

- Grid Technologies
- P2P Technologies

Cooperation / Communication

- Job management on P2P Grid system with Grid middleware
  - i.e. United Device’s Grid MP + GT3
- How is it going now?
- The same architecture as Community Scheduler Framework (CSF)
  - Platform’s LSF + GT3
Summary

- There are activities combining P2P and Grid technologies
  - They can be classified to several categories:
    - Grid over P2P
    - P2P over Grid
    - Cooperation (?)
  - This way of classification is not very sophisticated
    - Other aspects?

- Taxonomy?
  - Identify elemental P2P/Grid technologies
  - Classify the implemented instances according to the pattern of the integration

- Other ways to combine / integrate the elemental technologies?