Realizing the e-Science Desktop Peer Using a Peer-to-Peer Distributed Virtual Machine Middleware.

Lei Ni, Aaron Harwood and Peter J. Stuckey
NICTA Victoria Labs, Australia,
Department of Computer Science and Software Engineering,
The University of Melbourne, Australia

1

Presented at the 4th International Workshop on Middleware for Grid Computing, Melbourne, Australia, 27, Nov., 2006.

Outline

- What is the proposed e-Science Desktop Peer and why.
- P2P-DVM, a prototype of e-Science Desktop Peer.
- Experiments and results.
- Conclusion and future work.

2

Introduction

- e-Science requirements.
- Today's Internet.
- The challenges of Internet based solution.

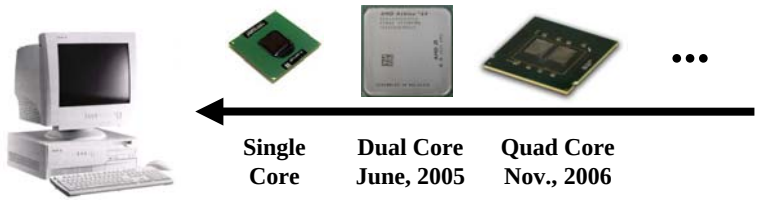
3

Motivation

- Decentralized architecture is preferred.
- The e-Science desktop peer that utilizes P2P techniques can solve the problem.

4

Motivation (cont.)



Relatively high performance desktops, e.g. multi-core processors

+

recent development in network technologies, e.g. P2P

↓

High performance while affordable computing.

5

Contributions

- Proposed concept of e-Science Desktop Peer.
- P2P-DVM, a grid middleware and prototype implementation of the e-Science Desktop Peer.
- The experiment results for our proposed P2P based architecture.

6

The e-Science Desktop Peer

- Runs on desktops. Vast amount of desktops on Internet. SETI@Home attracts about 1M PCs.
- Utilizes P2P techniques to build or integrate decentralized services for parallel processing.
  - Message Passing.
  - Data Storage.
  - Fault Tolerance.
- Share some similarities to the desktop grid but they are still different.

7

e-Science Desktop Peer Prototype

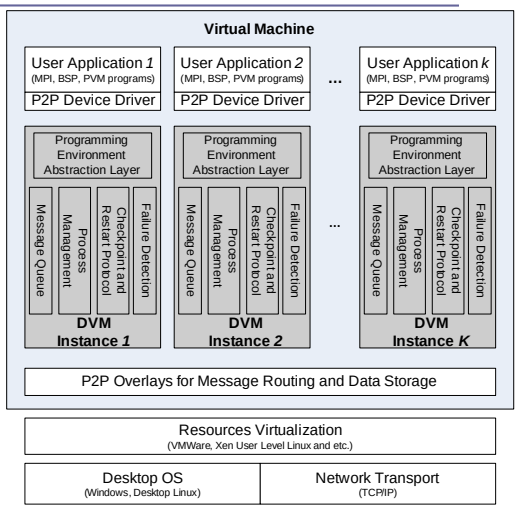
Multiple programming environments support

(MPI, BSP and etc.).

Decentralized process management, data storage, fault tolerant.

Decentralized communication.

Managed, isolated and guaranteed environment using virtualization.



8

P2P-DVM Architecture.

Virtualization in P2P-DVM

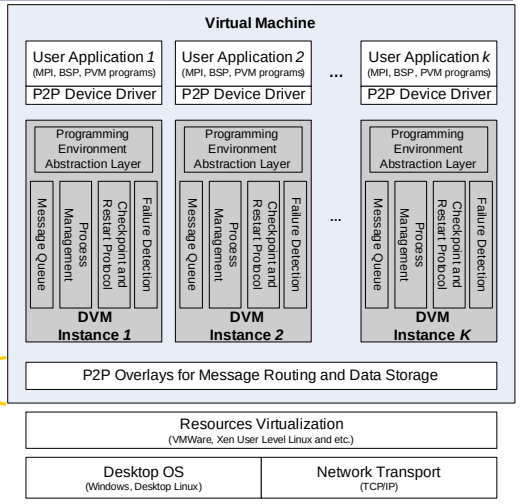
- Provides a managed, isolated and guaranteed environment.
- Also for security and privacy concerns.
- Different technologies are available out of the box and are almost transparent to the reset part of the system.

9

P2P-DVM Architecture

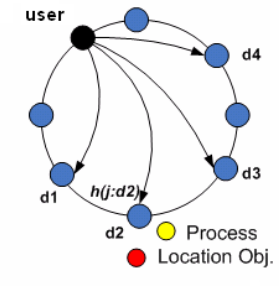
Decentralized communication.

(Floc protocol)



10

Decentralized Communication



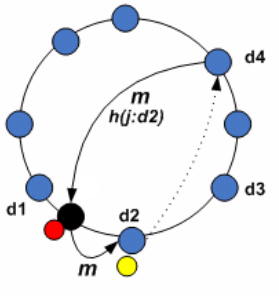
When submitting a new job.

Example:

1. When submitting a new job that requires 4 processes, the user peer sends out 4 spawn messages using the key h(j:d1), h(j:d2), h(j:d3), h(j:d4), where d1 to d4 are the identifier of processes, j is the job's identifier and h is the SHA-1 hash function.
2. On receiving such spawn message, both a location p2p object and the process will be created on the peer. The location object contains the physical location info of the process and it will migrate to its neighbours when the hash space change.
3. Now if any process want to communicate process di, the message can be routed to di using the key h(j:di).
4. Where the process will be spawned is by random.

11

Decentralized Communication (cont.)

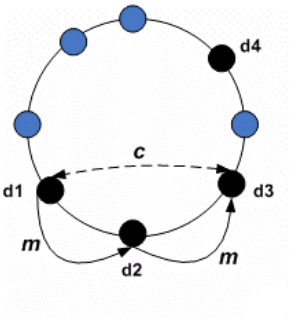


Reliable communication with the help of the ***location object***.

1. A new peer (the black one) join the P2P network. It changes the hash spaces.
2. If d4 want to send a message to d2, a key h(j:d2) will be used to route the message. As the hash space has changed, the message may be received by a peer that join the network after the job is submitted.
3. The P2P protocol ensures the location object has migrated to that peer and thus it is possible for that peer to re-route the message to the correct destination.

12

Floc Protocol



Shortcuts will be built between peers that communicate a lot.

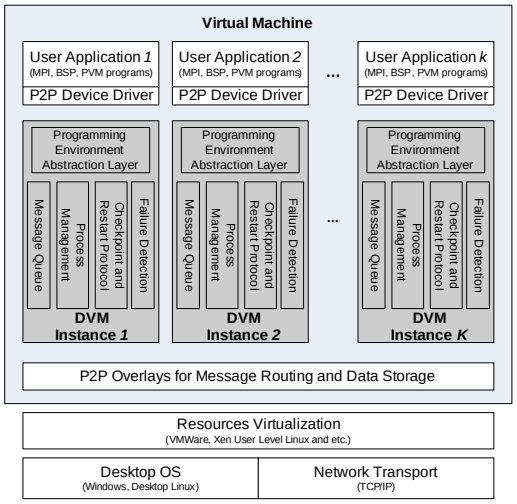
1. In a P2P network, it takes multiple P2P hops for each message to be sent to its destination. This can cause high latency for communication.
2. The Floc P2P protocol used in our system will build shortcuts between peers that communicate a lot and thus reduce the latency.

13

P2P-DVM Architecture

Multiple programming environments support.

Decentralized process management, data storage, fault tolerant.



14

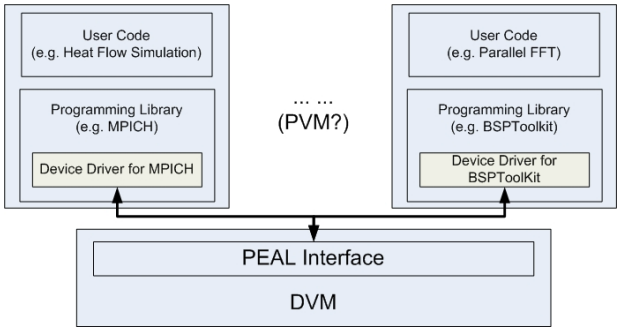
The Distributed Virtual Machine

- Job and process management, FIFO communication.
- The Programming Environment Abstraction Layer (PEAL), the interface for user programs.
- Fault tolerant protocol, reliable execution over unreliable P2P network.
- The Peer Client Interface allows users to interact with the DVM.

15

The PEAL Interface

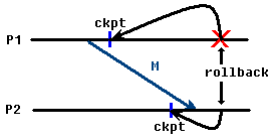
- High level abstraction of the common interface of different programming environments.
- Make it easier to support new environment, e.g. PVM.



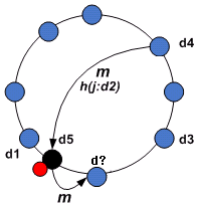
16

Fault Tolerant Support

- Decentralized coordinated checkpoint and restart.



- Decentralized checkpoint image storage. (Similar to the CFS)
- Decentralized failure detection.



17

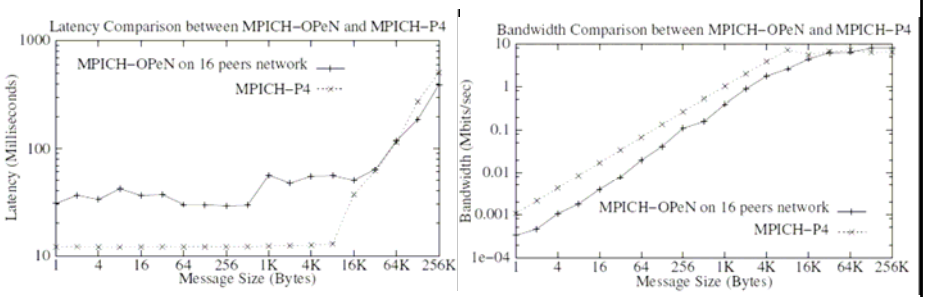
Experiments and Results

- PlanetLab is our test-bed.
- 16 Internet2 connected nodes from 9 US cities.
- Virtualization is provided by Linux VServer.
- Try to demonstrate the performance of services provided by our e-Science Desktop Peer prototype.

18

Message Passing

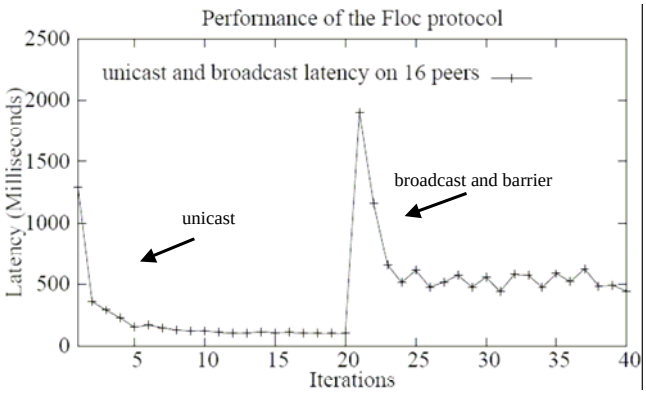
- Latency and bandwidth performance with different packet sizes using NetPIPE. We compare the results with MPICH-P4.



19

Message Passing (cont.)

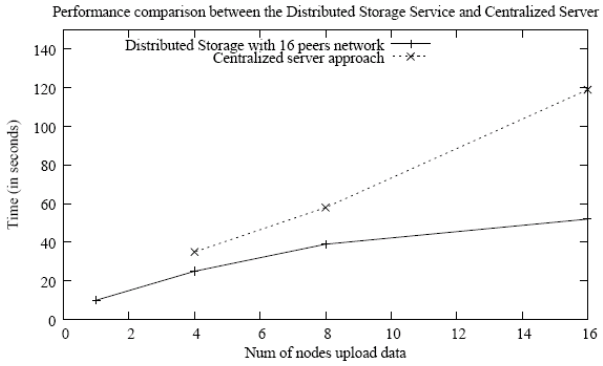
- Floc will quickly adapt to different communication patterns.



20

Decentralized Storage

- Performance comparison between our decentralized storage and centralized server when accepting files from multiple nodes.



21

The Overhead of Check Pointing

- The overhead of checkpointing in terms of total runtime and the size of the image need to be uploaded.
- Heat flow simulation program in MPI, using a 1024x1024 matrix as the input.

	4 procs	6 proces	12 procs
Runtime without checkpoint	546s	504s	581s
Runtime with checkpoint	604s	562s	771s
Overhead	10.6%	11.9%	32.7%
Checkpoint Image size	18.5m	11.5m	6.5m

22

Conclusion and Future Work

- The challenges that e-Science applications face.
- P2P based decentralized architecture can help to solve the problem.
- Working on parallel protein structural alignment, testing the application with our P2P-DVM.
- A larger scale deployment of the software.

23

Thank You! Questions?

For more information about our research projects:

http://www.cs.mu.oz.au/p2p

24