PeerSpaces: Data-Driven Coordination on P2P Networks (2003)

From Nadia Busi, Cristian Manfredini, Alberto Montresor and Gianluigi Zavattaro of University of Bologna

Presented by Samuele Milani

Summary

- New coordination model (The Peer Spaces Model)
- Exploit the Linda Model for Peer-to-Peer Networks.
- Prototypical implementation (JXTA).

Overview

1) Peer-to-Peer Networks
2) The Linda Model
3) Linda in a Mobile Environment (LIME)
4) Peer Spaces
5) An Implementation with JXTA

1. Peer-to-Peer Networks

Peer-to-Peer Basics

- Various type of Networks:
  - Client-Server: Web servers
  - Hybrid P2P: Napster, eDonkey
  - Pure P2P: Gnutella, Freenet

- a) Client/Server
- b) Hybrid P2P
- c) Pure P2P
The eDonkey Network

---

The Linda Model

- Developed at Yale University in 1985
- Shared memory accessible by all processes
- Data elements consist of tuples
- Tuple is like a record ex. ('worker', 0, 1)
- Four basic primitives
- Implemented through libraries (Linda for C, Fortran, C++)

---

2. The Linda Model

---

The Fantastic 4

- `out('test', 4, 64)` places a data tuple in tuple space
- `eval('test', i, f(i))` creates a live tuple, which start a process that evaluates each argument
- `rd('test', ?, ?, ?) blocking` read the values in a tuple in a tuple space
- `in('test', ?, ?, ?) blocking` same as rd except that it also removes the tuple from the tuple space

---

Example

```
Tuple Space

Boss

Slave

Slave
```

---
3. Lime
(Linda in Mobile Environment)
4. Motivation

- Adapt the Linda/Lime Model to P2P
- Generative communication
- Must support extreme dynamism in structure, content and load
- Large degree of self-configuration and self-management is required
- New coordination model PeerSpaces
- Formal rules, abstraction of data and network structure

4. A Peer Space Example

Peer p:
p(write(job, Gen).read(response, 8).write(cash, Gen).P, { })

Peer q:
q(read(job, 8).write(response, Rep).read(cash, 8).
  write(pay, Gen).Q, {dog})

Peer r:
r(read(pay, 8).write(receipt, Gen).R, {credit})

Network:
- fully connected
MD = {}
<table>
<thead>
<tr>
<th>Peer p:</th>
<th>p(read(response, 8), write(cash, Gen), P, {job})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peer q:</td>
<td>q(read(job, 8), write(response, Rep), read(cash, 8), write(pay, Gen), Q, {dog})</td>
</tr>
<tr>
<td>Peer r:</td>
<td>r(read(cash, 8), write(cash, Gen), R, {credit})</td>
</tr>
</tbody>
</table>

Network:  
- fully connected  
MD = \{\}

---

<table>
<thead>
<tr>
<th>Peer p:</th>
<th>p(read(response, 8), write(cash, Gen), P, {})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peer q:</td>
<td>q(read(job, 8), write(response, Rep), read(cash, 8), write(pay, Gen), Q, {dog})</td>
</tr>
<tr>
<td>Peer r:</td>
<td>r(read(cash, 8), write(cash, Gen), R, {credit})</td>
</tr>
</tbody>
</table>

Network:  
- fully connected  
MD = \{job\}

---

<table>
<thead>
<tr>
<th>Peer p:</th>
<th>p(read(response, 8), write(cash, Gen), P, {})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peer q:</td>
<td>q(read(job, 8), write(response, Rep), read(cash, 8), write(pay, Gen), Q, {dog, job, response})</td>
</tr>
<tr>
<td>Peer r:</td>
<td>r(read(cash, 8), write(cash, Gen), R, {credit})</td>
</tr>
</tbody>
</table>

Network:  
- fully connected  
MD = \{\}

---

<table>
<thead>
<tr>
<th>Peer p:</th>
<th>p(read(response, cash), P, {response, cash})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peer q:</td>
<td>q(read(cash, 8), write(pay, Gen), Q, {dog, job, response})</td>
</tr>
<tr>
<td>Peer r:</td>
<td>r(read(cash, 8), write(cash, Gen), R, {credit})</td>
</tr>
</tbody>
</table>

Network:  
- fully connected  
MD = \{\}
A Peer Space Example

Peer p:
p(P, {response})
Peer q:
  \texttt{write}(\texttt{pay}, \texttt{Gen}, Q, \{\texttt{dog, job, response, cash}\})
Peer r:
  \texttt{r}[(\texttt{read}(\texttt{pay}, B). \texttt{write}(\texttt{receipt}, \texttt{Gen})]. R, \{\texttt{credit}\})

Network:
- fully connected
MD = {}
Open issues?

- Implementation of:
  - Route (p, p')
  - Hor (p, h)
  - LoadBal (p, p'; d)

What's JXTA?

- JXTA is an open-source project by Sun to provide a set of basic facilities for P2P applications
- Based on a set of XML protocols
- JXTA middleware has 3 layers
  - core: low-level functions (routing, communication, ...)
  - services: indexing, searching, file sharing
  - applications: high-level application (chat, auction, ...)

Conclusion

- We cannot yet draw conclusions until concrete implementations come out !!!
- But the model should theoretically work

To be continued...