Self-Adaptive Discovery Mechanisms for Improved Performance in Fault-Tolerant Networks

Kevin Mills, Doug Montgomery, Scott Rose, Stephen Quirolgico, Kevin Bowers, and Chris Dabrowski

DARPA FTN PI Meeting
January 27, 2003

Scalable Software for Hostile & Volatile Environments
Self-Adaptive Discovery Mechanisms for Improved Performance in Fault-Tolerant Networks

Presented at DARPA Fault Tolerant Network (FTN) PI Meeting, 27 Jan 2003
Presentation Outline

- One-Page Review of Project Objective and Plan
- One-Page Refresher on Service Discovery Protocols
- Analysis of Jini Leasing Performance
- Self-Adaptive Leasing for Jini
  - Two Algorithms: Simple Adaptive Leasing and Inverted Leasing
  - Performance characteristics (*obtained via simulation*)
- Leasing with Multiple Lookup Services
- Summary of Other Accomplishments Since July 2002
- Plan for Next Six Months
- Conclusions
**Project Objective**


**Project Plan – Three Phases**

- **Phase I** – characterize performance of selected service discovery protocols (Universal Plug-and-Play – UPnP – and Jini) as specified and implemented
  - develop simulation models for each protocol
  - establish performance benchmarks based on default or recommended parameter values and on required or most likely implementation of behaviors

- **Phase II** – design, simulate, and evaluate self-adaptive algorithms to improve performance of discovery protocols regarding selected mechanisms
  - devise algorithms to adjust control parameters and behavior in each protocol
  - simulate performance of each algorithm against benchmark performance
  - select most promising algorithms for further development

- **Phase III** – implement and validate the most promising algorithms in publicly available reference software
**Dynamic Discovery Protocols in Essence**

Dynamic discovery protocols enable *network elements*:

1. to *discover* each other without prior arrangement,
2. to *express* opportunities for collaboration,
3. to *compose* themselves into larger collections that cooperate to meet an application need, and
4. to *detect and adapt to changes* in network topology.

### Selected First-Generation Dynamic Discovery Protocols

<table>
<thead>
<tr>
<th>Design</th>
<th>Network-Dependent 3-Party Design</th>
<th>Network-Dependent 2-Party Design</th>
<th>Adaptive 2/3-Party Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>JINI 3-Party Design</td>
<td>UPnP 2-Party Design</td>
<td>SLP</td>
<td></td>
</tr>
<tr>
<td>Vertically Integrated 3-Party Design</td>
<td>HAVi 3-Party Design</td>
<td>Bluetooth 2-Party Design</td>
<td></td>
</tr>
</tbody>
</table>
A Brief History of Leases in Distributed Systems


- Now widely used in distributed systems
  - Mobile Networking
  - Distributed File Systems
A Brief History of Leases in Distributed Systems (cont.)

- **Shared Memory**

- **Web Systems**

- **Service Discovery Systems**
Selected Jini Leasing Sequences

(a) Initial Lease Grant & Renewal

- Jini Service
  - $T$
  - Lease Request ($L_R$)
  - Lease Grant ($L_G \leq L_R$)
  - Lease Request ($L_R$)
  - Lease Grant ($L_G \leq L_R$)
  - Lease Cancel
  - Lease Cancelled

(b) Lease Denial

- Jini Service
  - $T$
  - Lease Request ($L_R > L_{MAX}$)
  - Lease Denied

(c) Lease Cancellation

- Jini Service
  - $T$
  - Lease Request ($L_R$)
  - Lease Grant ($L_G \leq L_R$)
  - Lease Cancel
  - Lease Cancelled

(d) Lease Expiration

- Jini Service
  - $T$
  - Lease Request ($L_R$)
  - Lease Grant ($L_G \leq L_R$)
  - Lease Expired

Bandwidth usage

Responsiveness

FAILURE

DETECTION
Analysis of Jini Leasing Performance

Let $N$ = number of leaseholders, $S_R$ = size of lease request message, and $S_G$ = size of lease grant message.

Bandwidth Consumption ($B$)

$$B = \left(\frac{N}{L_G}\right) \cdot (S_R + S_G)$$

Responsiveness ($R$)

$$R = \frac{L_G}{2}$$

Given requirements for $B$ and $R$, what lease period should be granted to each leaseholder and how many leaseholders can be supported?

$L_G = 2R$ \hspace{1cm} Re-writing responsiveness equation yields a value for lease period to grant

$N_{MAX} = \left(\frac{B \cdot L_G}{S_R + S_G}\right)$ \hspace{1cm} Transforming bandwidth equation indicates maximum system capacity

What decisions must the lease grantor make to guarantee $R$ and $B$?

1. Deny lease requests that would consume excessive bandwidth (i.e., when $L_R < L_G$)
2. Grant lease periods no greater than $L_G$ to ensure desired responsiveness
3. Deny lease requests when the number of leaseholders would exceed capacity (i.e., when $N = N_{MAX}$)
Simulation Results: Responsiveness and Bandwidth Usage vs. Network Size for Various $L_G$ Values

- $R (L_G = 120 \, s)$
- $R (L_G = 60 \, s)$
- $R (L_G = 15 \, s)$
- $B (L_G = 15 \, s)$
- $B (L_G = 60 \, s)$
- $B (L_G = 120 \, s)$
A Simple Adaptive Leasing Mechanism

**Goal:** limit bandwidth usage to $B$ and guarantee a minimum responsiveness ($R_{MIN}$), while achieving the best possible responsiveness $R > R_{MIN}$ when $N < N_{MAX}$

**Preliminary Analysis**

\[ L_{MAX} = 2R_{MIN} \]  
Minimum Responsiveness determines maximum granted lease period

\[ G = \frac{B}{S_{R} + S_{G}} \]  
Available bandwidth determines maximum lease renewals per second ($G$)

\[ L_{MIN} = \frac{1}{G} \]  
Assuming minimum system size of 1, $G$ determines minimum granted lease period

\[ L_{MIN} = 2R_{MAX} \]  
However, $1/G$ might place too great a load on the leaseholder, so instead choose a maximum responsiveness and let that determine the minimum granted lease period

$L_{MIN} \leq L_{G} \leq L_{MAX}$  
Vary the granted lease period within this range, using the following algorithm

Adaptive Algorithm for Varying $L_{G}$

- set \( L_{G} = \frac{N}{G} \);
- if \( L_{G} > L_{MAX} \) then deny the lease;
- elseif \( L_{G} < L_{MIN} \) then set \( L_{G} = L_{MIN} \);
- endif
- endif

1/29/2003
An Inverted Leasing Mechanism

Main Idea: lookup service periodically polls leaseholders on a multicast channel – adapting the polling interval to accommodate variations in the number of leaseholders.

Polling Interval

Each poll contains the interval \(D\) over which the lookup service listens for responses and an additional time \(A\) within which the next poll will be sent, and

\[ L_{MIN} \leq D + A \leq L_{MAX} \]

Additional Constraint

Grant leases only up to \( N \leq N_{MAX} \), where \( N_{MAX} = L_{MAX} \times G \).
Adaptive Algorithm for Inverted Leasing Mechanism

Adaptive Algorithm for Varying $D$ (and $A$) and Selecting $T_{POLL}$

set $D = \text{Max } (N / G, L_{MIN})$;
set $A = 0.2 \times D$
if $D + A > L_{MAX}$
    then set $A = 0$;
endif
set $T_{POLL} = \text{time} + D$;

Since $N \leq N_{MAX}$,
$D \leq L_{MAX}$

Preliminary Analysis

$B = S_P + \left( \frac{N}{P} \right) \cdot \left( S_{PR} + S_{RC} \right)$
where $P$ is the polling interval ($D \leq P \leq D + A \leq L_{MAX}$), $S_P$ is poll size, and $S_{PR}$ and $S_{RC}$ are size of poll response and confirm.

Assuming half of failures occur before poll and half after: $R = 1/2 \cdot (D/2) + 1/2 \cdot (3D/2) = D$
So inverted leasing will be only $1/2$ as responsive as simple adaptive leasing (recall $R = L_G / 2$)

From this result we can also infer that $R_{MAX} = L_{MIN}$ and $R_{MIN} = L_{MAX}$
Bandwidth Usage and Control Variable Value vs. Increasing Network Size for Adaptive and Inverted Leasing Algorithms
Bandwidth Usage and Responsiveness vs. Decreasing Network Size for Adaptive and Inverted Leasing Algorithms
Adaptive Leasing with Multiple Lookup Services

**Main Goal:** Given a domain-wide limit for leasing resources, expressed either in terms of bandwidth ($B_D$) or renewals per second ($G_D$), the main goal is to allocate a fair share of the resources to each lease grantor within the domain.

- Let $N_D$ represent the number of lookup services within a domain.
- Assume each Jini lookup service is configured with a network-wide resource budget for leasing (either $B_D$ or $G_D$).
- Each lookup service can compute its share of available resources (either $B_D / N_D$ or $G_D / N_D$).

But we need a means to increase and decrease the allocation of resources with changes in $N_D$.

- Jini facilitates monitoring $N_D$ by requiring each lookup service to announce itself periodically (every 120 s recommended) on a designated multicast channel.
- Each lookup service can increment $N_D$ when a new lookup service is heard and can decrement $N_D$ when an expected announcement is missed.

As $N_D$ varies, each lookup service can continuously adjust its share of the available domain-wide leasing resources.
Other Accomplishments Since July 2002

- Produced three papers

- Completed characterization of UPnP and Jini behavior in a tactical (multiple sensor-actuator) application during node failure

- Completed scalable (up to 500 nodes) discrete-event simulation model of the Service Location Protocol (SLP)
Plan for the Next Six Months

- Implement our simple self-adaptive leasing in the publicly available Jini reference code distributed by Sun Microsystems – and demonstrate it at the 2003 DARPA Information Survivability Conference and Exposition (DISCEX III) in April (we will show this demonstration again at the summer 2003 FTN PI meeting)

- Characterize the behavior of the service location protocol (SLP) under hostile and volatile conditions – expect a journal paper in 2003 characterizing the performance of Jini, UPnP, and SLP in response to power failure, communication failure, message loss, and node failure

- Formalize a generic model of service-discovery architectures, including structure, behavior, and properties – expect a journal paper in 2003

- Develop an analytical model of the consistency maintenance behavior of Jini and UPnP during communication failure – expect a journal paper

- Investigate self-adaptive mechanisms for inclusion in SLP

- Continue interactions with the Sun Microsystems Jini team; with Microsoft, Intel and the UPnP Forum; and with the IETF SLP group
Conclusions

- Emerging industry discovery protocols exhibit performance characteristics that vary based on parameter settings, network size, and resource availability.

- Tuning such dynamic systems cannot rely on manual configuration methods.

- We illustrated one case – Jini leasing – where values for granted lease periods interact with system size to determine performance and resource usage.

- We proposed two self-adaptive algorithms for Jini leasing, and we investigated relative performance of the algorithms.

- We explained how the simple adaptive leasing algorithm could be used in a Jini system with multiple lookup services.

- We believe that our simple adaptive leasing algorithm can also be used for UPnP event subscriptions and for SLP service registrations (with some adjustments).

- We have shared our findings with Sun, Microsoft, Intel, the UPnP Forum, and the IETF SLP group.