Wireless Networks of Devices: Resource Discovery

William Adjie Winoto, Elliot Schwartz, Anit Chakraborty, Jeremy Lilley, Hari Balakrishnan, John Guttag

MIT Lab for Computer Science
http://wind.lcs.mit.edu/

November 1999
Application: Location-dependent wireless services

- Spontaneous networking
- Automatically obtain map region & discover devices/services and people therein
- Access, control services and communicate with them
- Handle mobility & group communication
- Locate other useful services (e.g., nearest café)

App should be able to conveniently specify a resource and access it.
Challenges

- Configuration
- Routing
- Discovery
- Adaptation
- Security & privacy

Dynamic, mobile environment with no pre-configured support for internetworking or service location.
Today

- Mostly static topology & services
- Deploying new services is cumbersome
- Applications cannot learn about network
- Failures are common
- High management costs
Resource discovery

• Why is this hard?
  - Dynamic environment (mobility, performance changes, etc.)
  - No pre-configured support, no centralized servers
  - Must be easy to deploy (“ZERO” manual configuration)
  - Heterogeneous services & devices

• Approach: a new naming system & resolution architecture using intentional naming
Design goals

**Expressiveness**
Names must be descriptive, signifying application intent.

**Responsiveness**
Name resolver must track rapid changes.

**Robustness**
System must overcome resolver and service failures.

**Easy configuration**
Name resolvers must self-configure.
Intentional name Resolvers (INR) form a distributed overlay to integrate resolution and message routing.
Intentional Naming System (INS) principles

- Names are intentional based on attributes
  - Apps know WHAT they want, not WHERE
- INS integrates resolution and forwarding
  - Late binding of names to nodes
- INS resolvers replicate and cooperate
  - Soft-state name exchange protocol with periodic refreshes
- INS resolvers self-configure
  - Form an application-level overlay network
INS service model

Self-organizing app-level overlay network formed based on performance

- Early binding
- Late binding
- Intentional anycast
- Soft-state name dissemination
- Set of names
- Query
- Self-organizing app-level overlay network
What’s in a name?

• Expressive name language (like XML)

• Resolver architecture decoupled from language

• Names are **descriptive**
  - Providers announce names

• Names are **queries**
  - Attribute-value matches
  - Range queries
  - Wildcard matches

```plaintext
[vspace = netgroup]
[department = arch-lab
  [state = oregon
    [city = hillsboro]]]
[resolution = 800x600]
[access = public]
[status = ready]
```

```plaintext
[vspace = thermometer]
[building = ne-43
  [room = *]]
[tempature < 62°F]
```

• Expressive name language (like XML)

• Resolver architecture decoupled from language

• Names are **descriptive**
  - Providers announce names

• Names are **queries**
  - Attribute-value matches
  - Range queries
  - Wildcard matches

```plaintext
[vspace = netgroup]
[department = arch-lab
  [state = oregon
    [city = hillsboro]]]
[resolution = 800x600]
[access = public]
[status = ready]
```

```plaintext
[vspace = thermometer]
[building = ne-43
  [room = *]]
[tempature < 62°F]
```
Responsiveness: Late binding

- Mapping from name to location(s) can change rapidly
- Integrate resolution and messaging to track change
  - INR resolves name by lookup-forward, not by returning address
  - \texttt{lookup(name)} is a route
  - Forward along route
- A name can map to one location (“anycast”) or to many (“multicast”)
Late binding services

- Intentional anycast
  - INR picks one of several possible locations
  - Choice based on service-controlled metric
  - Overlay used to exchange names

- Intentional multicast
  - INR picks all overlay neighbors that "express interest" in name
  - Message flows along spanning tree
  - Overlay used to transfer data
Robustness: Names as soft-state

- Resolution via network of replicated resolvers
- Names are weakly consistent, like network-layer routes
  - Routing protocol to exchange names
- Fate sharing with service, not INRs
  - Name unresolved only if service absent
- Soft-state with expiration is robust against service/client failure
  - No need for explicit de-registration
Self-configuring solvers

- INRs configure using a distributed topology formation protocol
- DSR (DNS++) maintains list of candidate and active INRs
- INR-to-INR “ping” experiments “link weights”
- Current implementation forms (evolving) spanning tree
- INRs self-terminate if load is low
Efficient name lookups

• Data structure

• Lookup
  – AND operations among orthogonal attributes
  – For values pick the ones satisfying the lookup

• Polynomial-time worst case
Scaling issues

- Two potential problems
  - Lookup overhead
  - Routing protocol overhead
- Load-balancing by spawning new INR handles lookup problem
- Virtual space partitioning handles routing problem
  - Just spawning new INR is insufficient
Virtual space partitioning

vspace=camera

vspace=5th-floor

Routing updates for each vspace
Delegate this to another INR
WIND Applications

- Location-dependent mobile applications
- Floorplan: A navigation & discovery tool
- Camera: An image/video service
- Printer: A smart print spooler
- TV & jukebox
- Location-support system based on intelligent beacons
<table>
<thead>
<tr>
<th>Rank</th>
<th>Owner</th>
<th>Job</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
<td>wang</td>
<td>6</td>
<td>/usr/tmp/figures.ps.Z3404</td>
</tr>
<tr>
<td>2nd</td>
<td>liuba</td>
<td>0</td>
<td>nested.ps, nested0.ps, nested1.ps</td>
</tr>
<tr>
<td>3rd</td>
<td>wadzie</td>
<td>29</td>
<td></td>
</tr>
</tbody>
</table>

**Printer Error:** may need attention! (Not Responding for 2 hours)

Floorplan: [org...]

10th floor
9th floor
8th floor
7th floor

Camera: 504

Update
Subscribe
Unsubscribe

LPR Client: pastrami @ room 537

Update queue list
Submit job to pastrami
Submit job to room 537
Remove job
Status & performance

- Java implementation of DNS applications
- PC-based resolver performance
  - 1 resolver: several thousand names @ 0-1000 lookups/s
  - Discovery time in linear hops
- Scalability
  - Virtual space partitions for load-shedding
  - Wide-area design in future
- Deployment
  - Hook in wide-area architecture to DNS
  - Standardize virtual place names (like MIME)
- Paper at SOSP 17
Related work

- Domain Name System
  - Differences in expressiveness and architecture
- Service Location Protocol
  - More centralized, less spontaneous
- Jini
  - INS can be used for self-organization and fault-tolerant discovery
- Universal Plug-and-Play & SSDP
  - XML-based descriptions; INS fits well
- Intentional names in other contexts
  - Semantic file systems, adaptive web caching, DistributedDirector
Future Internet Architecture

Use each other to add value

Middleware

Cache & replica management
Self-configuring overlays
Media transcoder
...

Performance discovery
Service location
E-speak

Resource management

Traffic engineering
Scheduling, buffer mgmt
Flexible IP routers

Decentralized security
Jini
UPnP
T-spaces

INS

Use each other to add value
Conclusion

• Achieving self-organizing networks requires a flexible naming system for resource discovery
  - INS works in dynamic, heterogeneous networks
  - Expressiveness: names convey intent
  - Responsiveness: late binding
  - Robustness: soft-state names
  - Configuration: Resolvers self-configure

• Application-level overlay networks are a good way to build flexible, self-organizing network applications