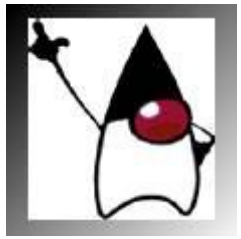
NEDERLANDSE JAVA USER GROUP

Jini and how further

Richard van der Laan

luminis

Agenda



- Concepts and architecture
- Jini community
- Technical roadmap
- Jini in OSGi context
- Sharing experiences
- Questions and your experiences

Background (1/2)

- **Richard van der Laan** (1977)
  - software engineer
  - distributed, object-oriented development
  - early adopter of Jini and OSGi technology
  - last years involved in change of architectural styles towards service-orientation

Background (2/2)

luminis®

- Professional services company
  - area of distributed systems
  - object-orientation (most Java)
  - innovative projects
- Combined with
  - management and process improvement
  - training & coaching on all levels of software development

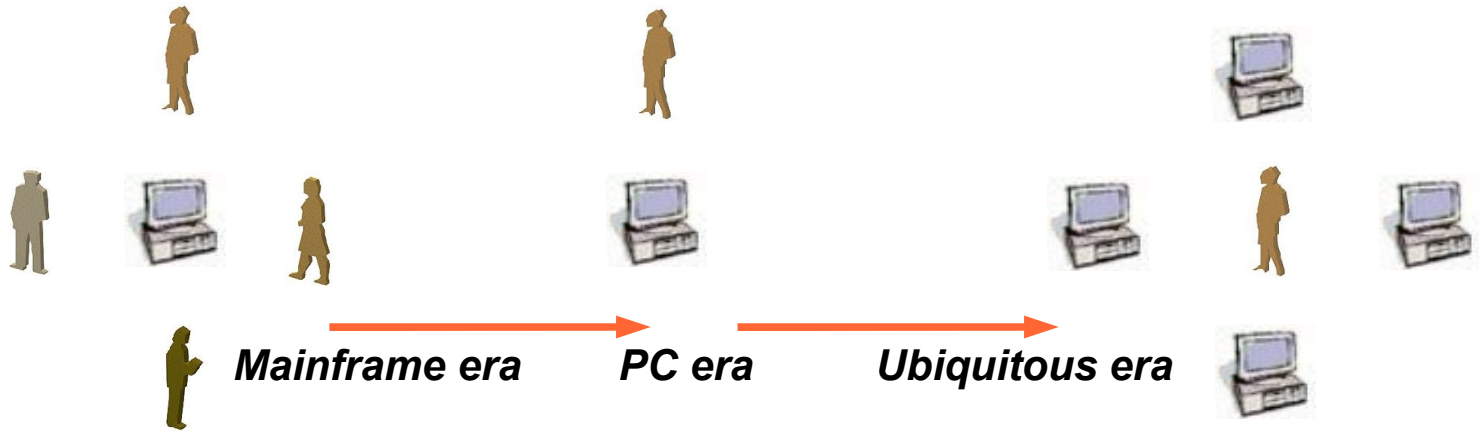
"Building the right system"
"Building the system right"
"Getting the system built"

Ubiquitous computing

"You don't want to use a computer (even though you think you do)"

Ubiquitous computing

- Many computers serve each person
- Any service, anywhere, anytime



Ubiquitous computing

- Focus on activity
- Emerging behavior
- Context aware devices

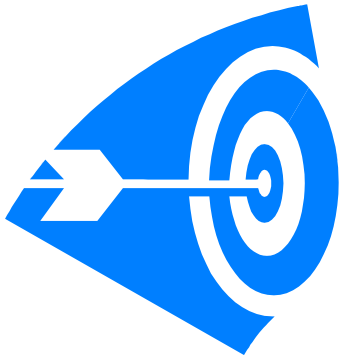
"I don't want to use a computer, I want to accomplish something"

What is Jini?

- Provides a flexible infrastructure for delivering distributed services
- Enables spontaneous interaction between services
- Adapts to changes in the network

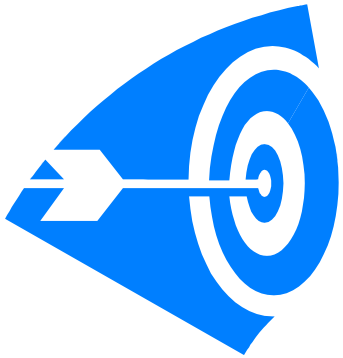
"Jini Is Not Initials"
Ken Arnold - Sun

Deal with the eight fallacies (1/2)



- The network is reliable
- Latency is zero
- Bandwidth is infinite
- The network is secure

Deal with the eight fallacies (2/2)



- Topology doesn’t change
- There is one administrator
- Transport cost is zero
- The network is homogeneous

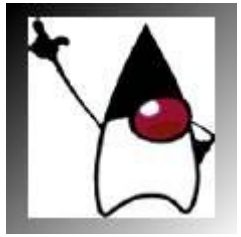
A Jini system consists of

- Infrastructural components
- Distributed programming model
- Services

Environmental assumptions

- Existence of a network
- Host has some memory and processing power
- Java technology centered

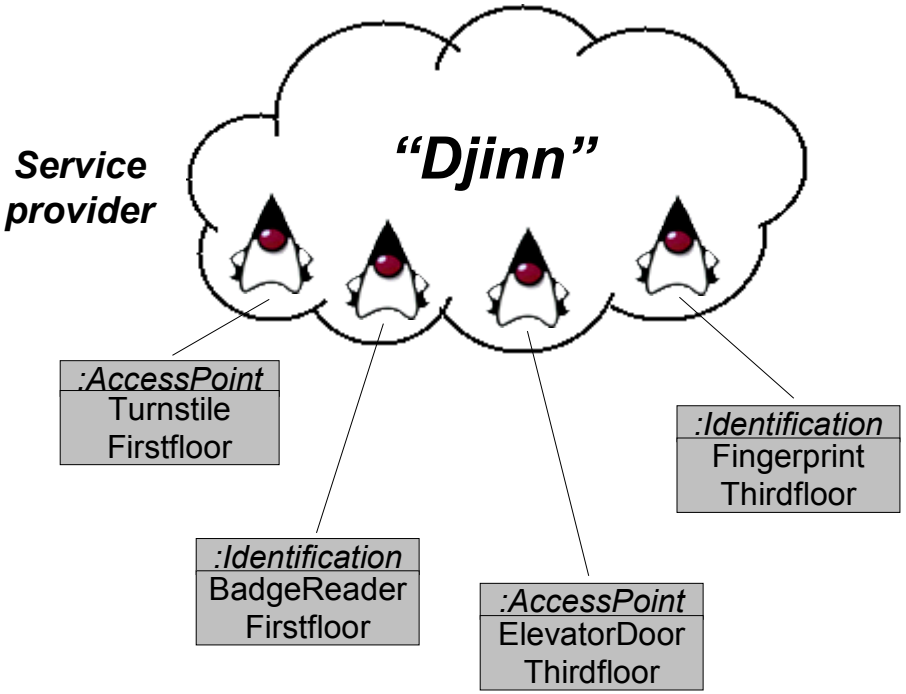
Key concepts



- *Discovery*
- *Lookup*
- *Leasing*
- Remote events
- Transactions

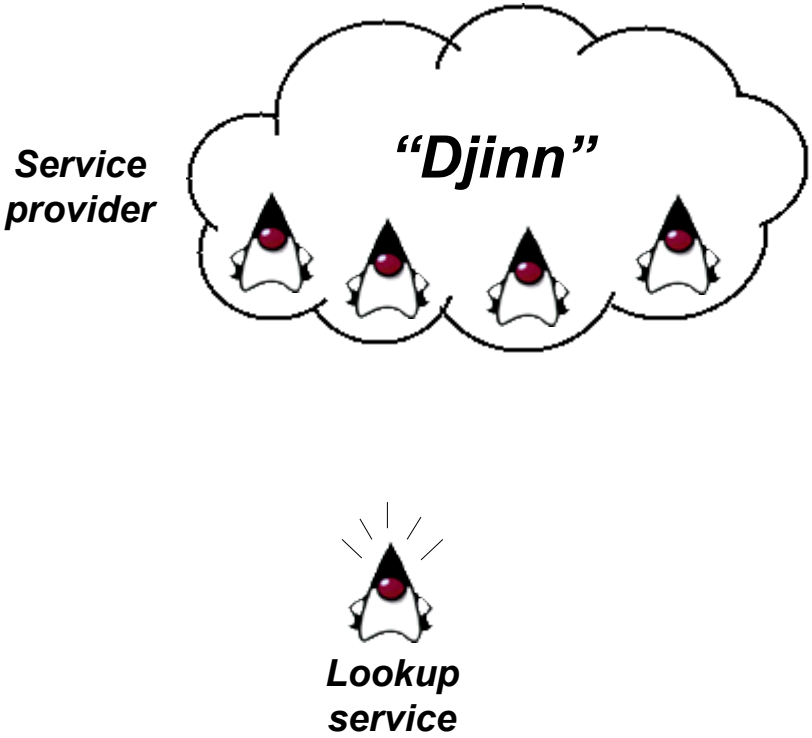
Discovery

- Services are federated in groups
- Services are described by interface and attributes



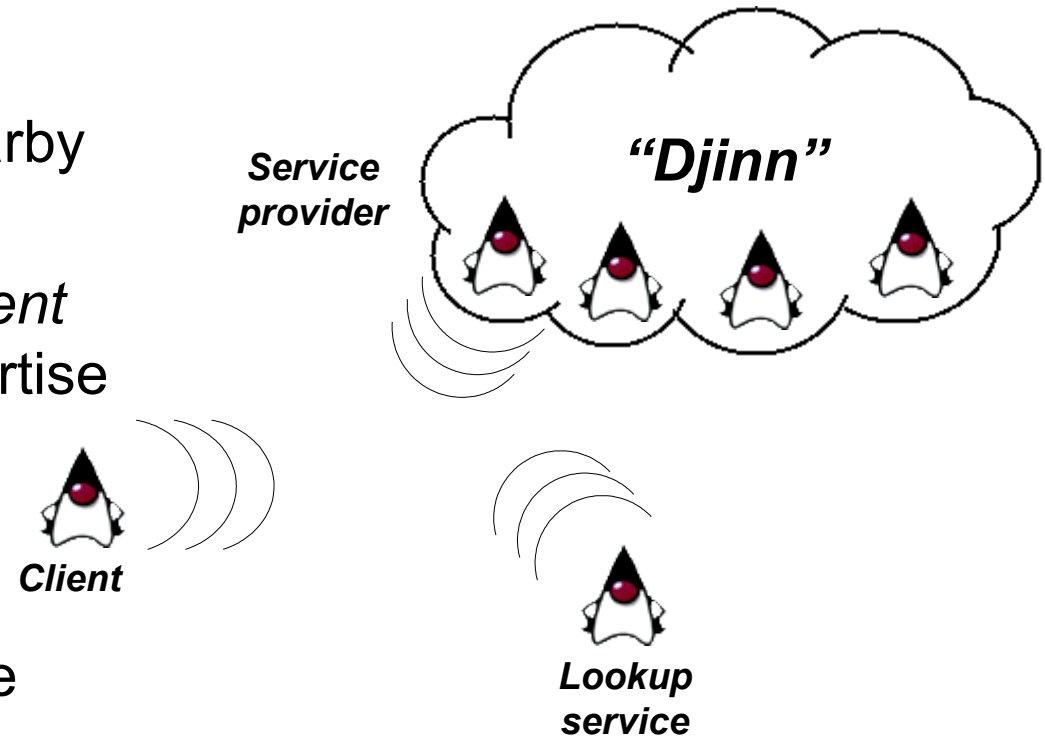
Discovery

- Lookup service acts as service broker
- Provide location transparency
- Deal with fail-over
- Enable clustering



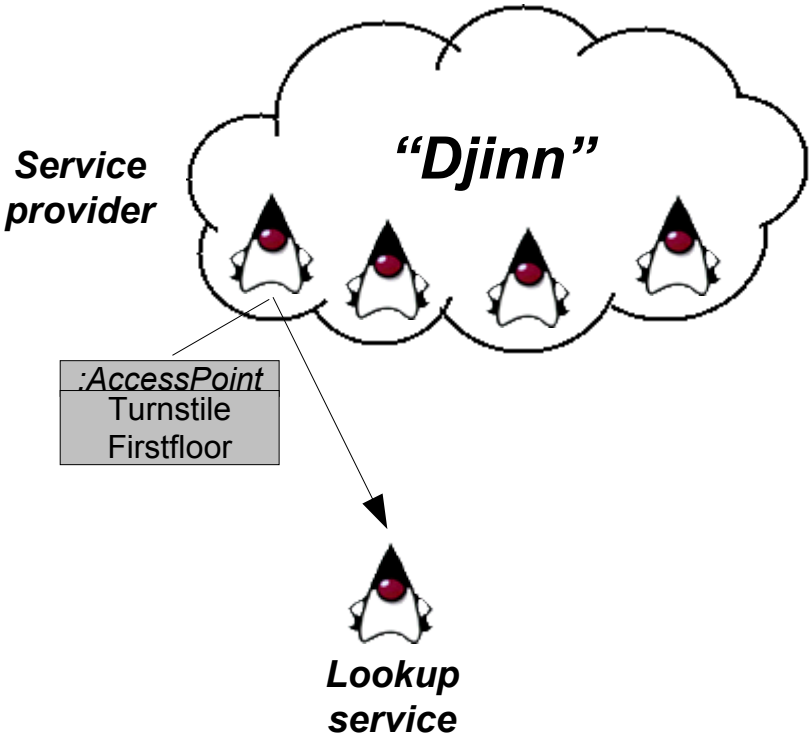
Discovery

- *Multicast request*
  Services discover nearby lookup services
- *Multicast announcement*
  Lookup services advertise their existence
- *Unicast discovery*
  Communicate with specific lookup service



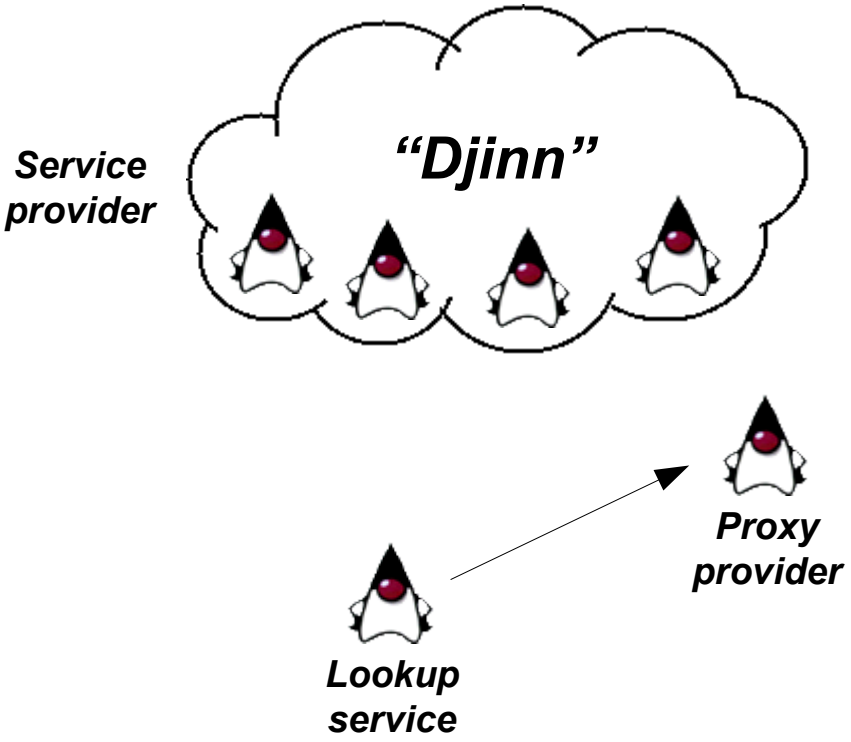
Discovery and join

- Service provider registers service object proxy with lookup service
- Proxy abstracts protocol
- Proxy can deal with system aspects such as:
  - security
  - fail-over / load-balancing
  - logging ..



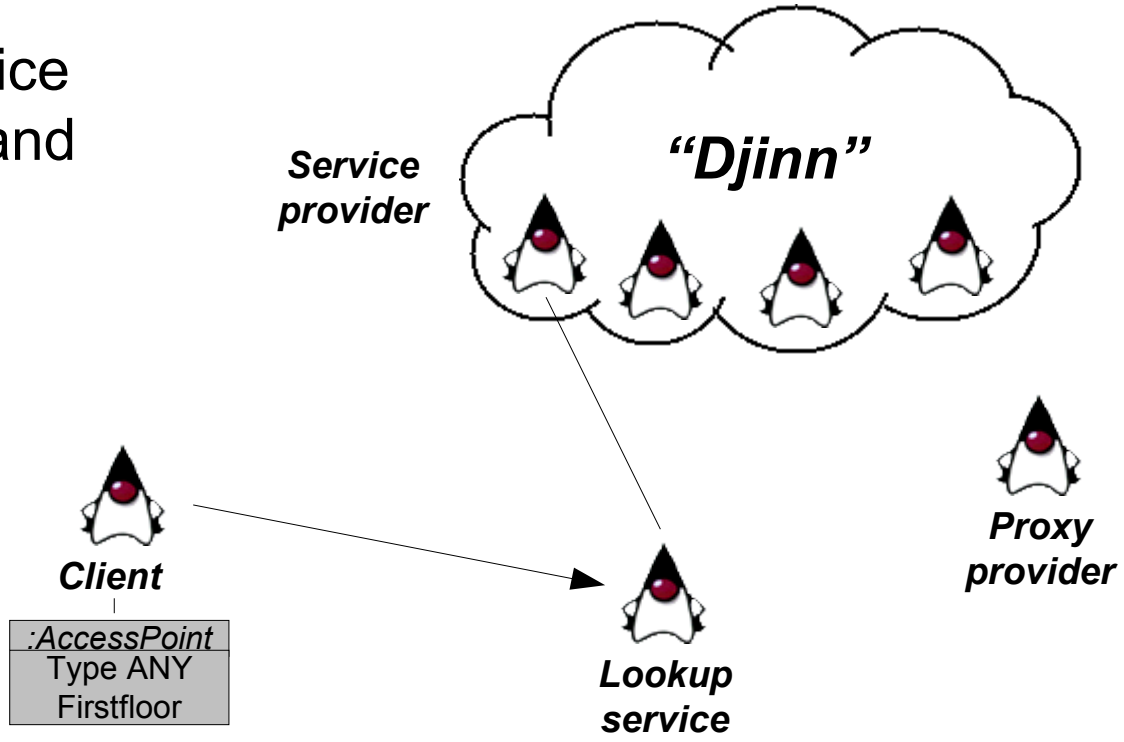
Discovery and join

- Unique service identification
- Download proxy bytecode
- Ability to move objects and code around network



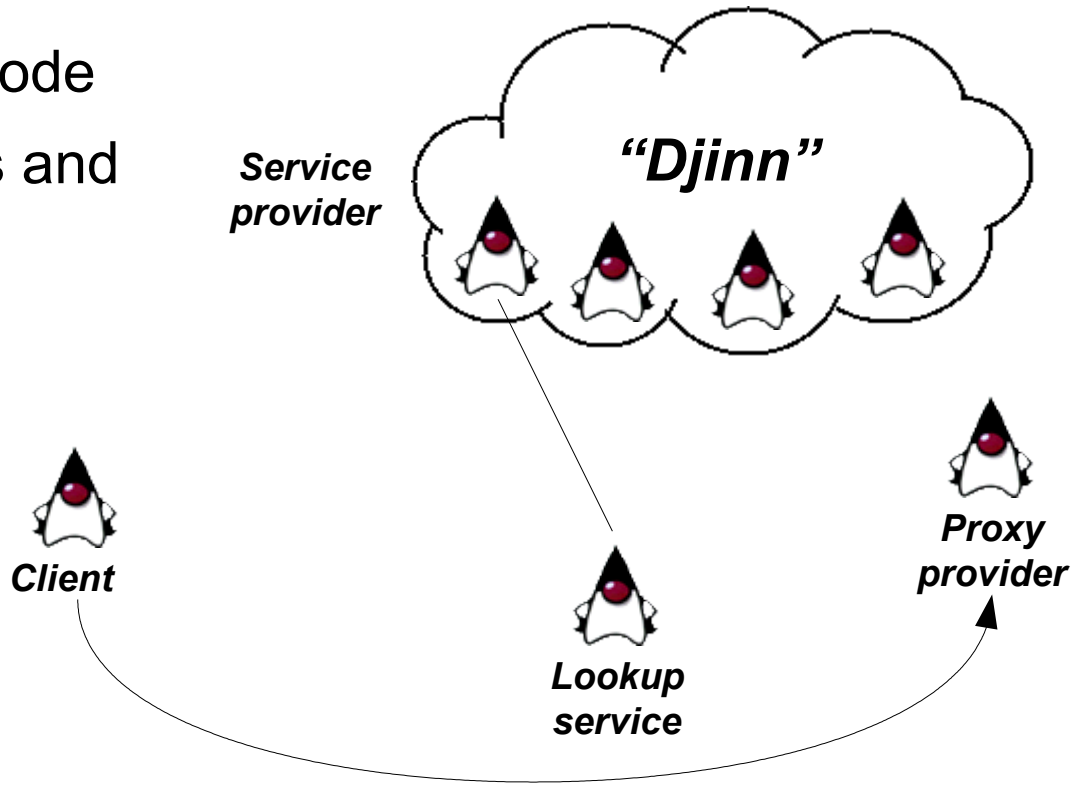
Lookup

- Client performs service lookup by interface and attributes



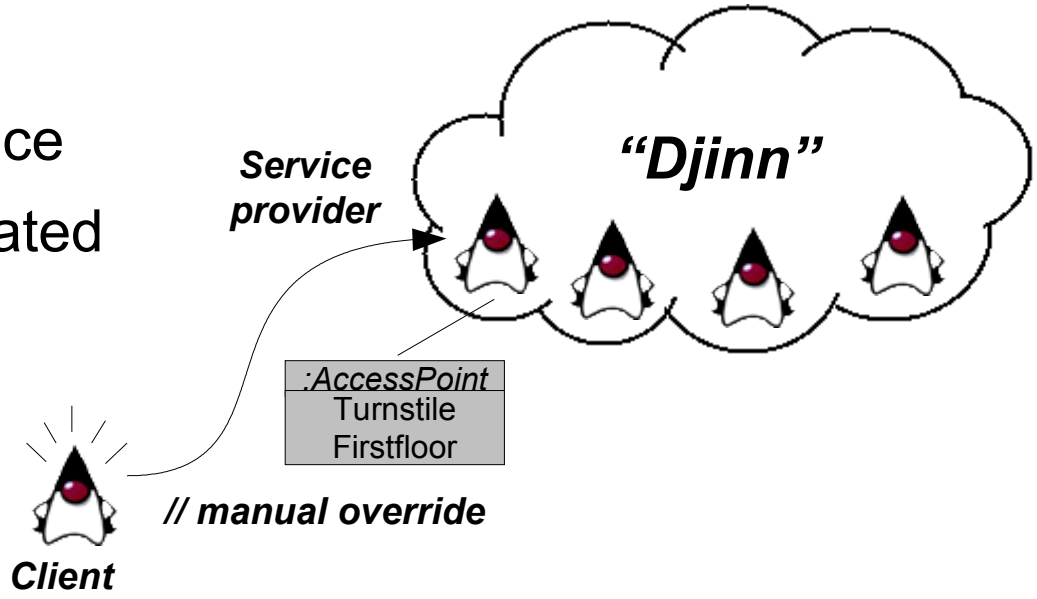
Lookup

- Download proxy bytecode
- Ability to move objects and code to client



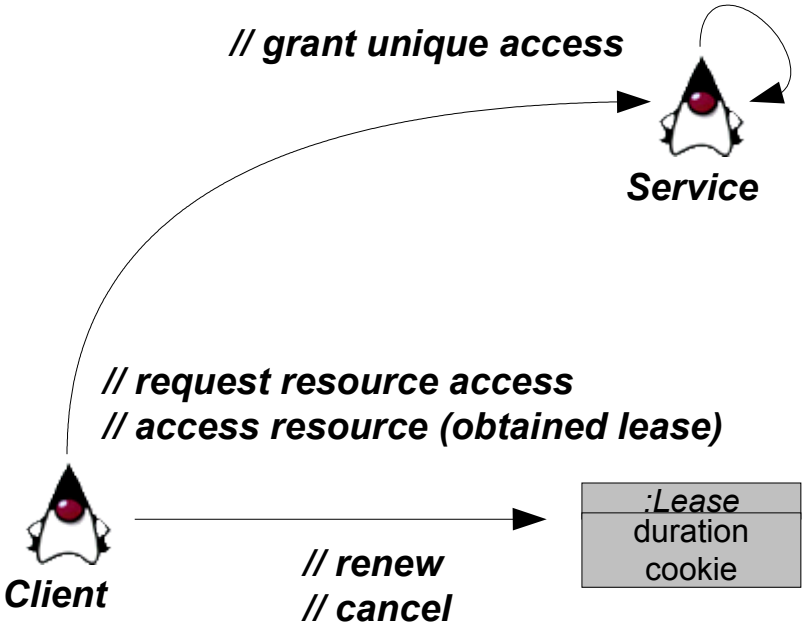
Lookup

- Client ready to access service through interface
- Communication delegated to proxy



Leasing

- Guaranteed access over a period of time
- Negotiation between service user and provider
- Exclusive or non-exclusive
- Deals with partial-failure



Contributed Jini services from Sun

- Lookup service (*Reggie*)
- Transaction manager (*Mahalo*)
- Lease renewal service (*Norm*)
- Event mailbox service (*Mercury*)
- Lookup discovery service (*Fiddler*)
- JavaSpaces service (*Outrigger*)

JavaSpaces

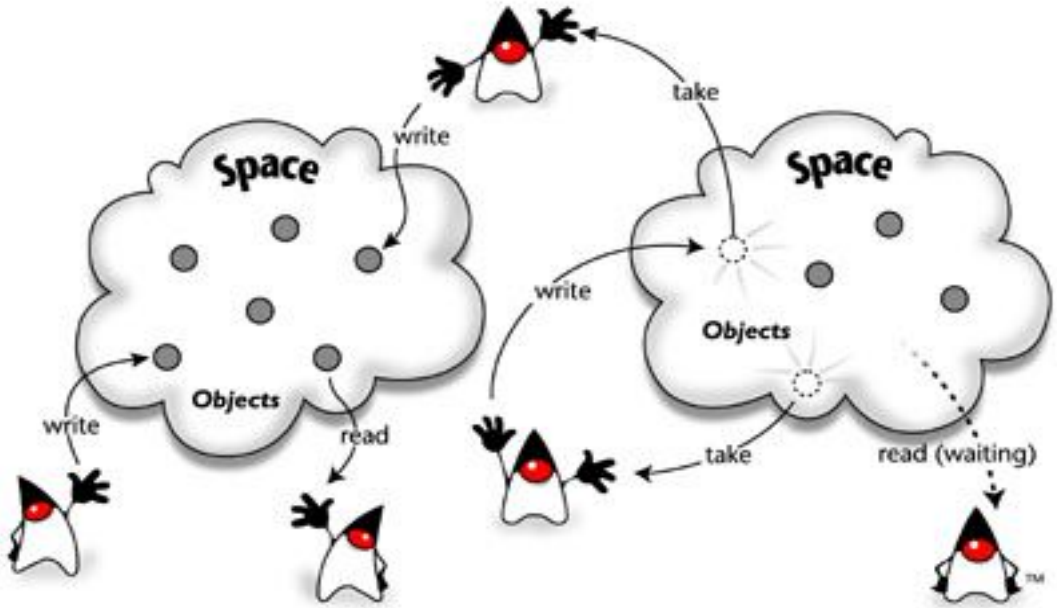
- Solve distributed computing issues in a easy and powerful way
- Based on Linda “tuple-spaces” (*David Gelernter*, Yale University)
- A virtual space between providers and requesters of resources and objects

JavaSpaces

- Provides a simple mechanism for
  - Communication
  - Coordination
  - Data and object sharing / persistency
- Entry leasing and transaction support
- Loose coupling

JavaSpaces

- Key methods:
  - write()
  - read()
  - take()
  - notify()



JavaSpaces

- Other available implementations:
  - AutevoSpaces
  - Blitz JavaSpaces (opensource)
  - GigaSpaces
  - TSpaces (IBM)

Jini community

- Established in January 1999
- Sun's gift to more collective ownership
- Standards approval by vote (JDP)
- Sun Community Source License (SCSL)

Jini community

- Many high quality non-Sun contributions
- Help each other and bring others in
- Annual community meetings
  (5th meeting dec. 2000 held in A'dam)

Roadmap

Beyond 2004
- Administration
- Pluggable datasources
- Webservices

1999 | *2000* | *2001* | *2002* | *2003* | *2004*

Jini
- Technology
- SW Kit v1.0

Jini
- Technology
- SW Kit v1.2
- Resource
- Efficiency

Jini
- Technology
- SW Kit v2.0 002
- Security
- Jini ERI

Davis Project
- Security
- Secure utility / service impl.
- Jini Extensible Remote Invocation
- Configurability

Jini Extensible Remote Invocation (ERI)

- Access to architectural layers
- Customizable framework
  - Invocation, dispatch behavior
  - Transport types
- Support for invocation constraints
  - Security

Jini security

- Downloaded proxy code
  - Verify that proxies can be trusted
  - Attach client constraints
  - Dynamically grant permissions
- Invocation constraints
  - Client / server authentication models
  - Message integrity and confidentiality

Jini work in progress

- Administration (Thor project)
  - Centralised configuration for distributed apps.
- Pluggable datasources (Athena project)
  - Universal service front-end with pluggable host connection to any datasource
- Webservices (Judy project)
  - Bridging the gap between Jini and web services

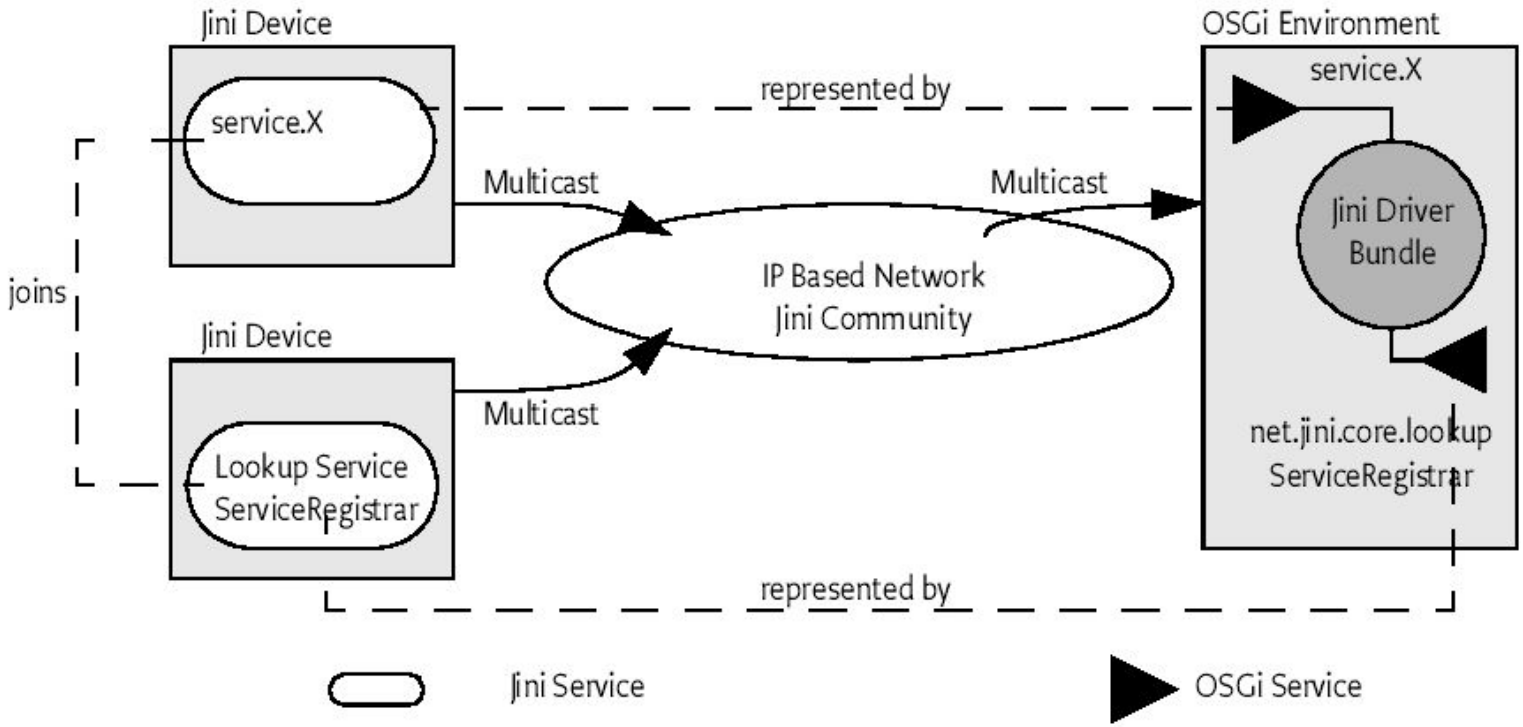
A broader context: OSGi

- Open Services Gateway Initiative (1999)
- Open common architecture to develop, deploy and manage services in a coordinated fashion
- Primary target products:
  - set top boxes, service gateways, cars ..

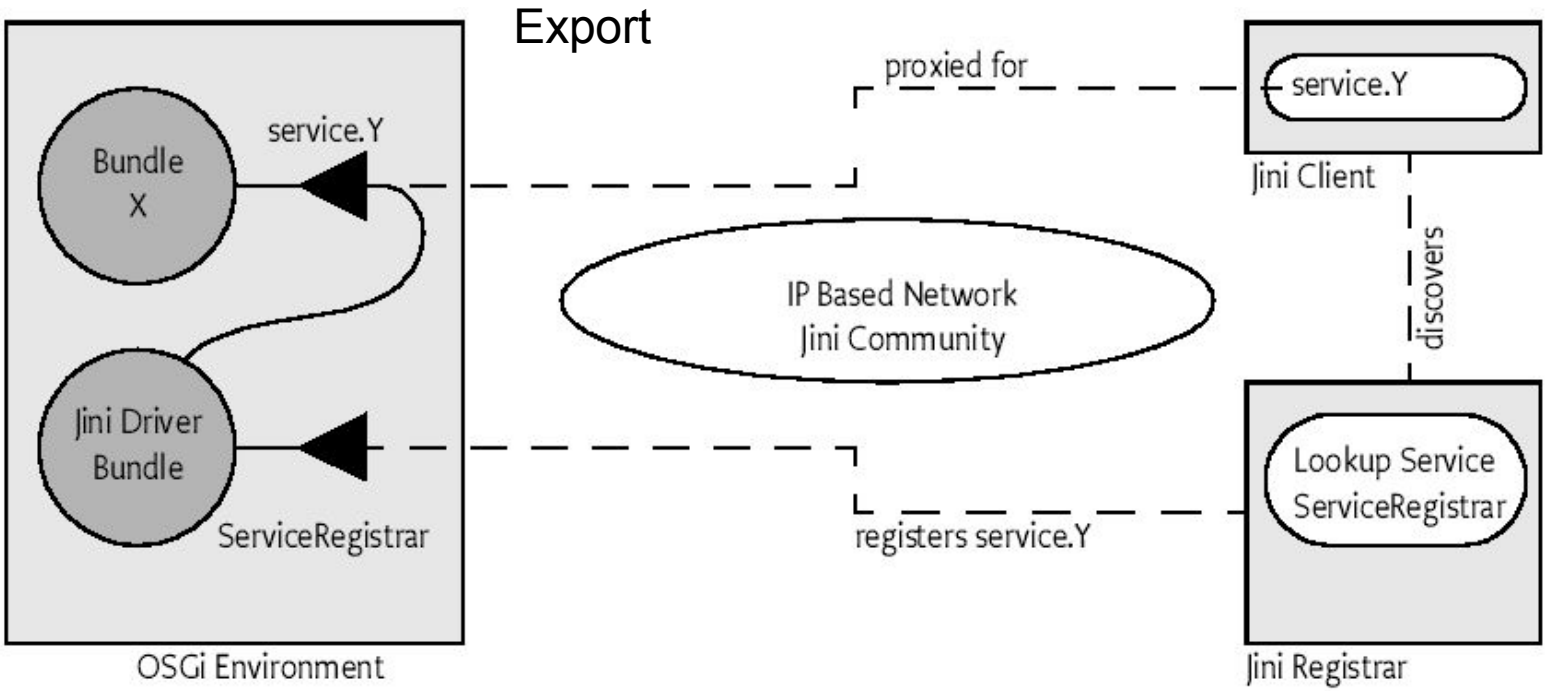
OSGi Jini Driver Service specification

- Discovery and control of Jini services within the framework
- Export OSGi services as Jini services

Importing Jini services



Exporting OSGi services



Current shortcomings

- No transparency
  - Lookup using Jini *ServiceTemplate* objects requires import and use of Jini driver service
  - Service exporting requires service to carry registration property *JiniDriver.EXPORT*
  - Service exporting requires service to implement *Serializable*
  - Proxies must be made available to clients

Current shortcomings

- No security
  - Imported services inherit Jini Driver permissions (running unchecked code)
  - Invocation of exported services simply bypass installed management agent
- No correlation between IO Connector
  - Providing an intermediate layer that abstracts actual used protocols

Sharing experiences

Nedap AEOS

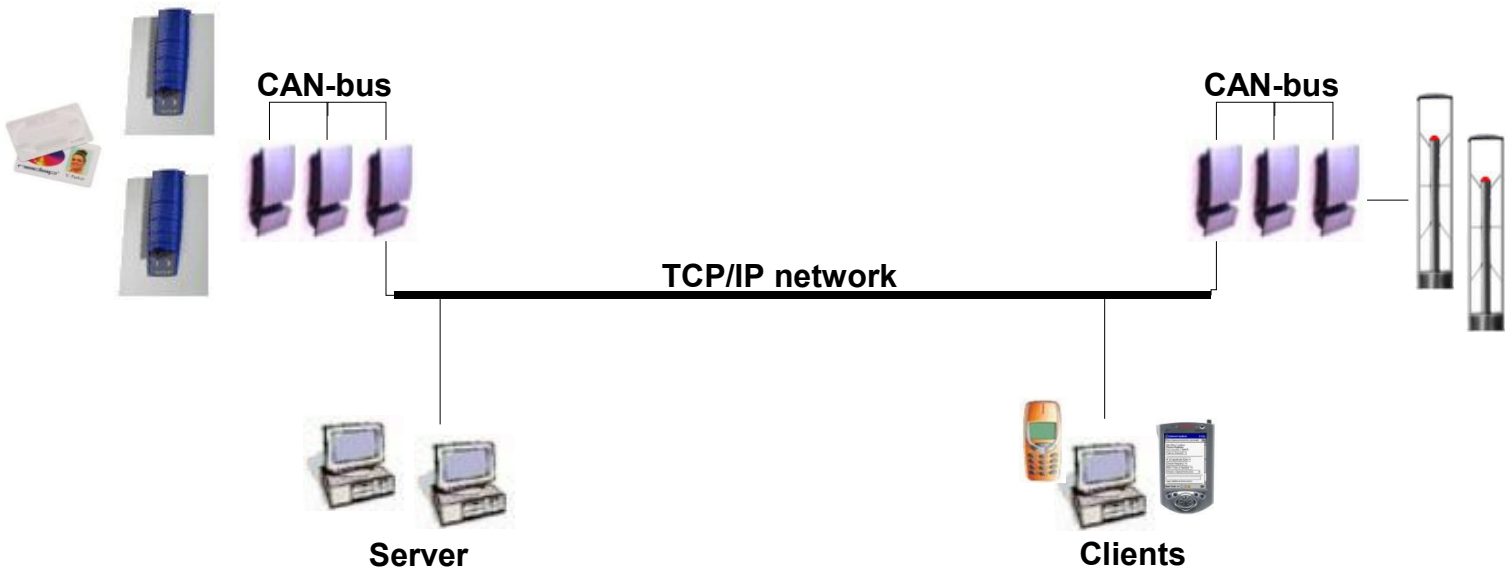
- Security management system
- Widely being used by airports, banks, government agencies

- Scalable, distributed architecture
- Freedom in identification
  RF-tag, iris, fingerprint, barcode ..

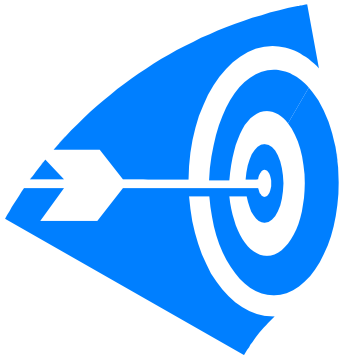
Nedap AEOS

- J2EE based
- Jini network services
  - Hardware primitives (actuators, sensors)
  - Security domain (identification, accesspoints)
- Plug 'n play device connectivity
- Web-enabled user interface

Nedap AEOS system overview



Sharing the good stuff



- Amazing technology
- Reliability through simplicity
- Open source (know how it works)
- Strong community feedback (bugfixes)

Sharing the bad stuff

- Hard learning curve for developers
- Distributed systems require a different approach and understanding
- Lack of tools for troubleshooting
- Integration problems (Enhydra, JBoss)
- Reference implementation bugs

Biggest misconceptions

- “Jini is primarily concerned with devices”
- “Jini is based on a proprietary network protocol”

Is Jini ready to leave the bottle?

- Growing community interest
- Change of architectural styles in commercial projects
- More applicability in embedded devices
- Serious attempts to leave the Java centric environment

NEDERLANDSE JAVA USER GROUP

www.luminis.nl
info@luminis.nl