Administrarea Bazelor de Date
Managementul în Tehnologia Informaţiei

Sisteme Informatice şi Standarde Deschise (SISD)

2009-2010

Curs 12

Tehnologii pentru sisteme Peer-to-Peer*

*Preluare și adaptare din cursul de LPD 2008-2009

Continut

- Sisteme Peer-to-Peer
- Jini
- Proiectul JXTA
- JavaSpaces

Retele peer-to-peer (P2P)

- Retele bazate pe puterea de calcul si pe latimea de banda a tuturor nodurilor componente (nu a unui numar limitat de servere)
- Nodurile sunt echivalente: functioneaza ca si “clienti”, dar si ca “servere”
- Utilizari:
  - Calcul distribuit (ex. SETI@Home)
  - Download distribuit (ex. BitTorrent)
  - Comunicare (ex. IRC)
  - Motoare de cautare
  - Colaborare
  - Securitate

Tipuri de retele P2P

- Traditionale (pure):
  - Nodurile au capabilitati si responsabilitati echivalente
  - Nu exista servere sau routere centrale
  - Tipuri:
    - “equal peer” – toate nodurile sunt perfect echivalente
    - “super peer” – unele noduri au roluri speciale
  - Exemple: Gnutella, Freenet, BitTorrent
- Hibride
  - Exista un server central care detine si ofera informatii despre noduri
  - Nodurile informeaza serverul central despre resursele detinute
  - Tipuri:
    - Centralizate (ex. SETI@Home, unele sisteme de chat)
    - Pe baza de brokering (ex. Napster)

Evolutia sistemelor P2P

- 1988: IRC
- 1999: Napster, Direct Connect
- 2000: Gnutella
- 2001: FastTrack
- 2001: BitTorrent
  - Trackers
  - Distributed hashtable
  - Web seeding
- 2004: estimare – 35% din traficul Internet e reprezentat de BitTorrent

Avantaje P2P

- Scalabilitate: resursele sistemului cresc o data cu cresterea numarului de noduri (clienti):
  - Latime de banda
  - Putere de calcul
  - Spatiu de stocare
- Toleranta la defecte: nu exista un “single point of failure”
- Cautarile se pot face in paralel, obtinandu-se un numar mai mare de rezultate

Continut

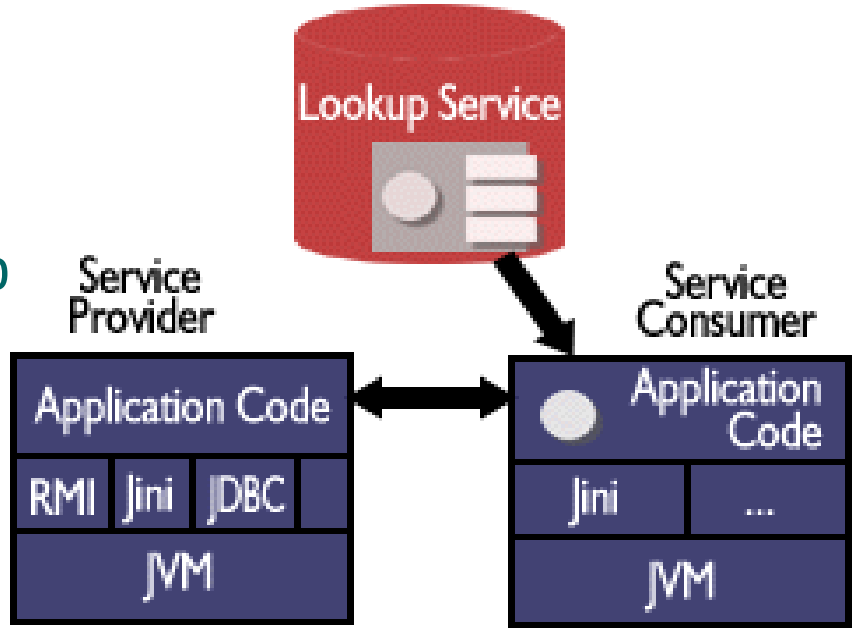
- Sisteme Peer-to-Peer
- Jini
- Proiectul JXTA
- JavaSpaces

Tehnologia Jini

- Tendinte noi in sistemele de calcul: importanta conexiunii la retea creste fata de importanta existentei unui disc
- Tehnologia Jini:
  - incercare de a restructura arhitectura sistemelor distribuite, tinand cont de rolul conexiunii la retea
  - set de API-uri si protocoale pentru realizarea de sisteme distribuite organizate ca federatii de servicii
  - bazata pe tehnologia Java, RMI si serializare
  - ofera mecanisme pentru adaugarea, stergerea si descoperirea serviciilor in mod dinamic
  - serviciile pot fi de orice tip, hardware sau software

Structura Jini

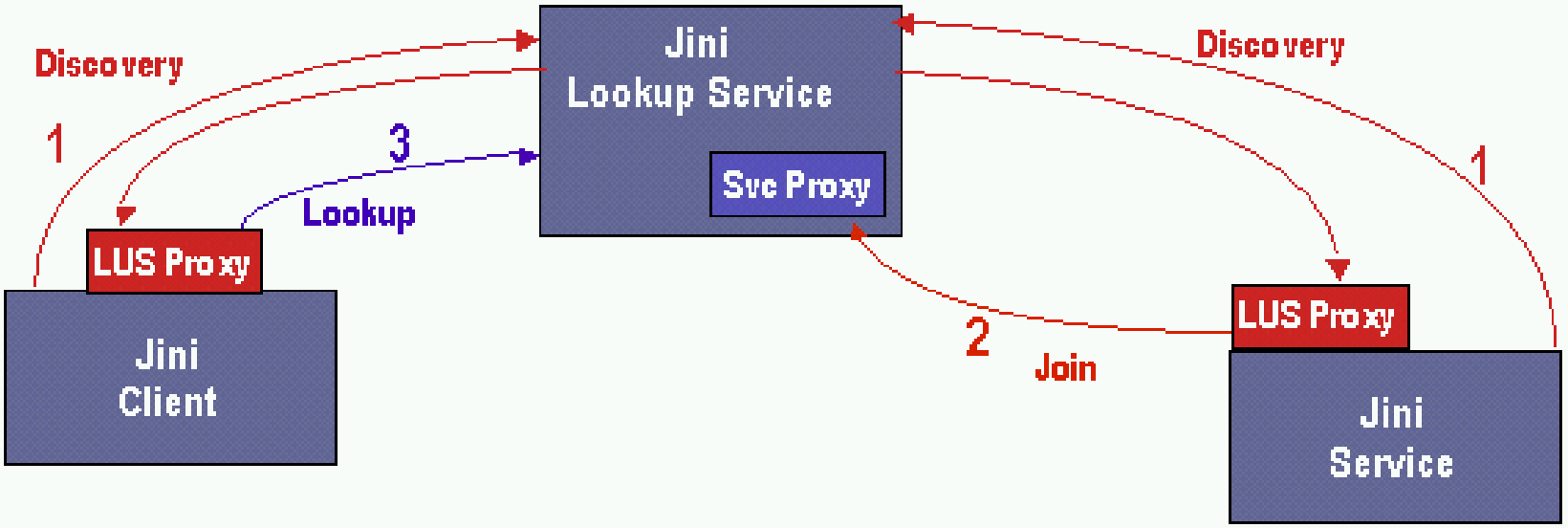
- Federatie de servicii
  - Nu exista un punct de control centralizat
  - Serviciile sunt organizate in maniera peer-to-peer
- Infrastructura Jini (runtime infrastructure) – ofera mecanisme pentru adaugarea, stergerea, localizarea si accesarea de servicii
  - *Lookup services*
  - Furnizori de servicii
  - Clienti
- Protocoale: discovery, join, lookup



Descoperirea serviciilor

- Protocolul *discovery*
- La pornirea unui serviciu: transmite prin multicast un pachet de anunt (anouncement) pe un port standard
- Serviciul de lookup monitorizeaza portul standard si, la primirea unui anunt, poate contacta serviciul pornit
- Serviciul de lookup transmite noului serviciu un obiect de tip *service registrar*, pe care serviciul il va utiliza in operatiile ulterioare
- Protocolul *discovery* este executat si de catre clienti, nu numai de catre servicii

Discovery, Join si Lookup in Jini [Kot04]



Jini networks track available services dynamically
Jini clients identify capable Jini services by Java type

Intrarea intr-o federatie de servicii

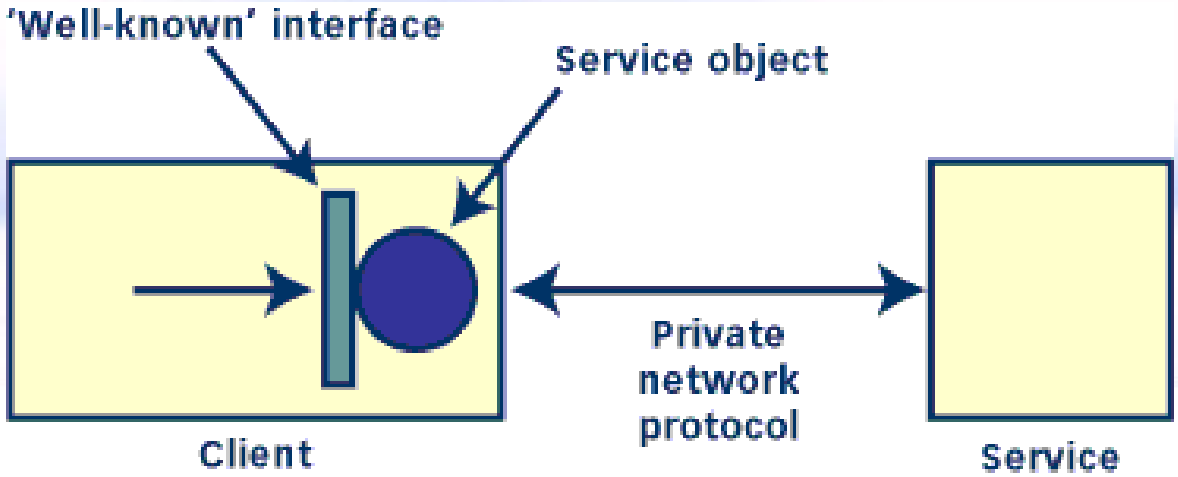
- Protocolul *join* – urmeaza dupa faza de *discovery*
- Serviciul nou a primit un obiect de tip `service registrar`
- Pentru intrarea in federatie: apeleaza metoda `register()` din `service registrar`
- `Service item` – parametru al metodei `register` – contine:
  - `service object` – obiect care va fi folosit de clienti pentru a invoca serviciul; implementeaza una sau mai multe interfete cunoscute de clienti
  - atribute (optional)

Cautarea unui serviciu

- Protocolul de *lookup*
- Clientul invoca metoda `lookup()` a obiectului `service registrar`
- `Service template` – argument al metodei `lookup()` – contine:
  - Un tablou de obiecte `Class`
  - Un `serviceID`
  - Atribute
- In template pot exista *wildcard*-uri

Comunicatia intre client si serviciu [Ven99]

- Clientul nu trebuie sa cunoasca protocolul de comunicatie!
- Protocolul este cunoscut doar de catre serviciu si de catre *service object*
- Se pot folosi RMI, CORBA sau alte protocoale



Rezervari distribuite (Distributed Leasing)

- “Negociere” privind timpul de utilizare a unei resurse
- Ocuparea unei resurse pentru un timp nedefinit nu este indicata in sistemele distribuite
- Lease: rezervarea pentru un timp finit
  - Poate fi anulata sau reinnoita
- Interfata de baza: `net.jini.Lease`

Diferente intre Jini si RMI

- Serviciul de lookup Jini poate fi descoperit si in mod dinamic (dar numai in reteaua locala)
- Serviciul de lookup Jini poate stoca orice obiect serializabil, registry-ul RMI poate stoca doar stub-uri RMI
- In Jini obiectele pot fi cautate si dupa tip, in RMI se poate cauta doar dupa nume
- Serviciul de lookup Jini este dinamic (serviciile sunt inregistrate pentru o perioada limitata de timp – lease)
- In Jini serviciile sunt organizate pe grupuri

Avantaje Jini

- Un service object poate functiona in mai multe moduri:
  - Operatia se executa in totalitate la client, in cadrul service object –ului primit dupa lookup
  - Service object este doar un proxy si operatia se executa la distanta pe un server
- Separarea interfetei de implementare: clientul nu trebuie sa cunoasca protocolul care exista eventual intre service object si server
- Implementarea serviciului se poate modifica in timp
- Nivel de abstractizare ridicat
- Diferitii furnizorii de servicii nu trebuie sa se puna de acord asupra protocolului de retea prin care acestea interactioneaza, ci doar asupra interfetelor Java

Continut

- Sisteme Peer-to-Peer
- Jini
- Proiectul JXTA
- JavaSpaces

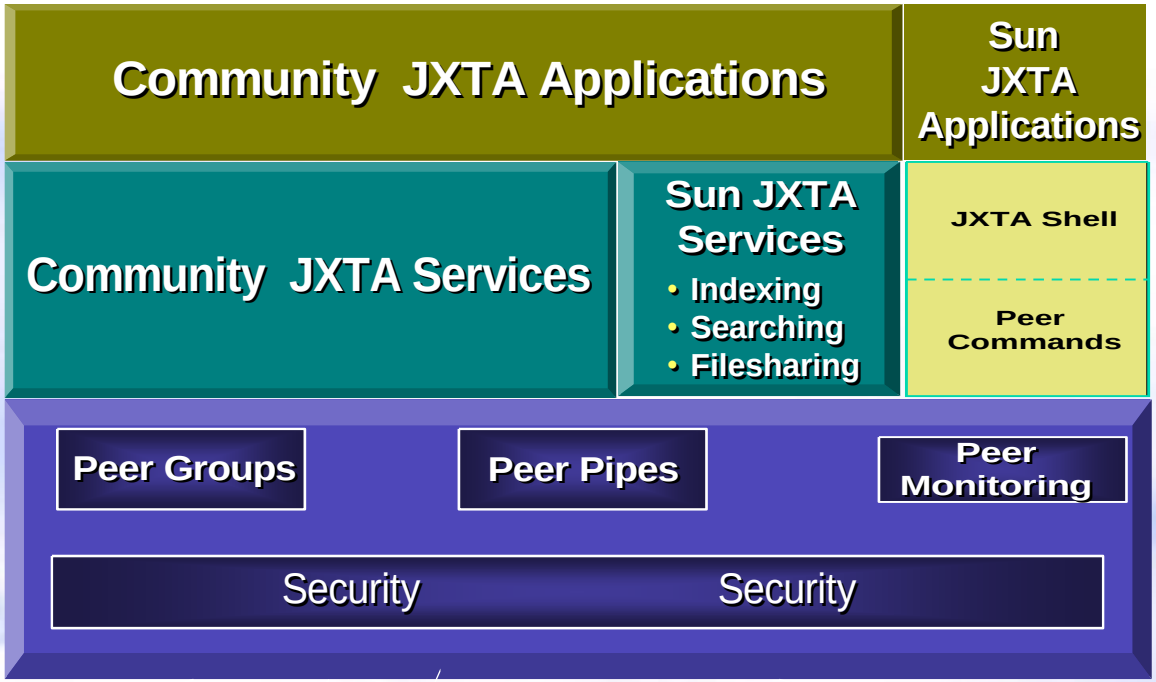
Proiectul JXTA

- Problema: lipsa de interoperabilitate intre sisteme P2P diferite
- Este necesar un “limbaj comun” pentru crearea aplicatiilor
- JXTA: set de specificatii de protocoale pentru aplicatii P2P
- Proiectat pentru a putea fi utilizat intr-un spectru cat mai larg de aplicatii:
  - Independenta de platforma (sistem de operare, limbaj de programare, protocol de transport)
  - Nodurile pot fi dispozitive digitale de orice tip, inclusiv sisteme embedded
  - Nodurile pot aparea/disparea in mod spontan din retea

Componentele unui sistem JXTA

- “Peers” si grupuri de “peers”
- Servicii
- Pipe-uri
- Mesaje
- Referinte

(advertisements)








Servicii de baza JXTA

- Pipe - principalul mod de comunicare intre “peers”; pipe-urile sunt unidirectionale, asincrone
- Membership – determina apartenenta la grupuri
- Access – serviciu de securitate care controleaza accesul la resurse in cadrul unui grup
- Discovery
- Resolver – permite identificarea unor entitati din sistem (“peers”, grupuri, servicii etc.) prin intermediul unor referinte

Stiva de protocoale JXTA

- **Peer Discovery Protocol** – pentru descoperirea serviciilor intr-un sistem P2P
- **Peer Resolver Protocol** – pentru transmiterea si procesarea de cereri generice
- **Rendezvous Protocol** – propagarea de mesaje intre noduri
- **Peer Information Protocol –** pentru obtinerea informatiilor de stare despre noduri
- **Pipe Binding Protocol** – pentru asocierea unui canal de comunicare cu un nod
- **Endpoint Routing Protocol** - pentru rutarea mesajelor intre noduri

Avantaje JXTA

- Interoperabilitate intre sistemele P2P
- Independenta de platforma
- Ofera un nivel inalt de abstractizare
- Utilizarea XML pentru comunicarea de mesaje – standardizare, mesajele sunt usor de citit

Dezavantaje JXTA

- Nu specifica modul de invocare a serviciilor
- Mesajele XML: overhead suplimentar (poate fi inacceptabil pentru unele aplicatii)
- Independenta de protocolul de transport: overhead suplimentar; este necesara, avand in vedere ca in marea majoritate a cazurilor se utilizeaza TCP/IP?

JINI vs. JXTA

- Ofera solutii diferite ale unor probleme similare
- JINI
  1. “federation of services & clients” prin transport de obiecte Java in retea
  2. depinde de existenta unui nod JAVA din retea
  3. JAVA API based
  4. IP networks based
  5. Java Security
- JXTA
  1. localizarea participantilor, acces la serviciile definite
  2. platform independent
  3. foloseste protocoale si este independenta de limbaj
  4. transport layer independent
  5. nu impune un anumit mecanism dezvoltatorului
  6. specifica formatul si continutul mesajelor schimbate
  7. poate detecta serciviul look-up JINI prin anunturi XML
  8. TLS transport security

Security

JINI

- Autentificare centrala prin Look-up service
- Access Control List
- Client-server security prin Java RMI

JXTA

- Implementeaza grupuri care necesita autentificare
- ID-uri unice pentru fiecare nod: pw & username criptate
- Mesaje sigure

Portability

JINI

- Cel putin un nod necesita platforma JAVA
- OS – independent
- Necesita IP

JXTA

- Platform independent
- Bazat pe protocoale independente de platforma
- Transmiterea mesajelor prin XML
- Network layer independent
- IP momentan

Robustness & Scalability

JINI

- Testat intesiv de piata
- Are cativa ani buni in spate
- Depinde foarte mult de Look-up service
- Scalabila

JXTA

- Nu a fost testat de piata prea mult
- Standarde gen XML si protocoale sigure
- No servers -> no central points of failures
- Probleme pentru grupuri mari (60 – 70 clienti)

JINI and JXTA for Agent Systems

- Suporta aplicatii in medii eterogene si dinamice
- Folosite pentru dezvoltarea aplicatiilor bazate pe agenti
- JINI: infrastructura de activare pentru CoABS
- CoABS
  - Co-operating Agent Based Systems
  - Toolkit pentru interfatarea mai multor toolkit-uri de agenti
  - “wrapping agents” + JINI service – API comun
  - Agentii se descopera folosind serviciile JINI si comunica prin interfata wrapper
- JINI: comunicatie prin RMI
  - Necesita un proxy
  - Comunicatie complicata intre agenti care nu sunt in aceeasi subretea
- JXTA: relativ nou, putine sisteme bazate pe agenti pe JXTA

Ce alegem? Jini sau JXTA?

- JINI – aplicatii care necesita mecanisme robuste pentru comunicare si descoperire, costul nefiind o problema
  - + functionalitati dezvoltatorului
  - + flexibilitate in folosirea diverselor tipuri de agenti
  - - dispozitive cu capacitati limitate

- JXTA – aplicatii care impun o comunicatie flexibila
  - + costuri reduse pentru administrare
  - - servicii limitate (trebuie proiectate)

Continut

- Sisteme Peer-to-Peer
- Jini
- Proiectul JXTA
- JavaSpaces

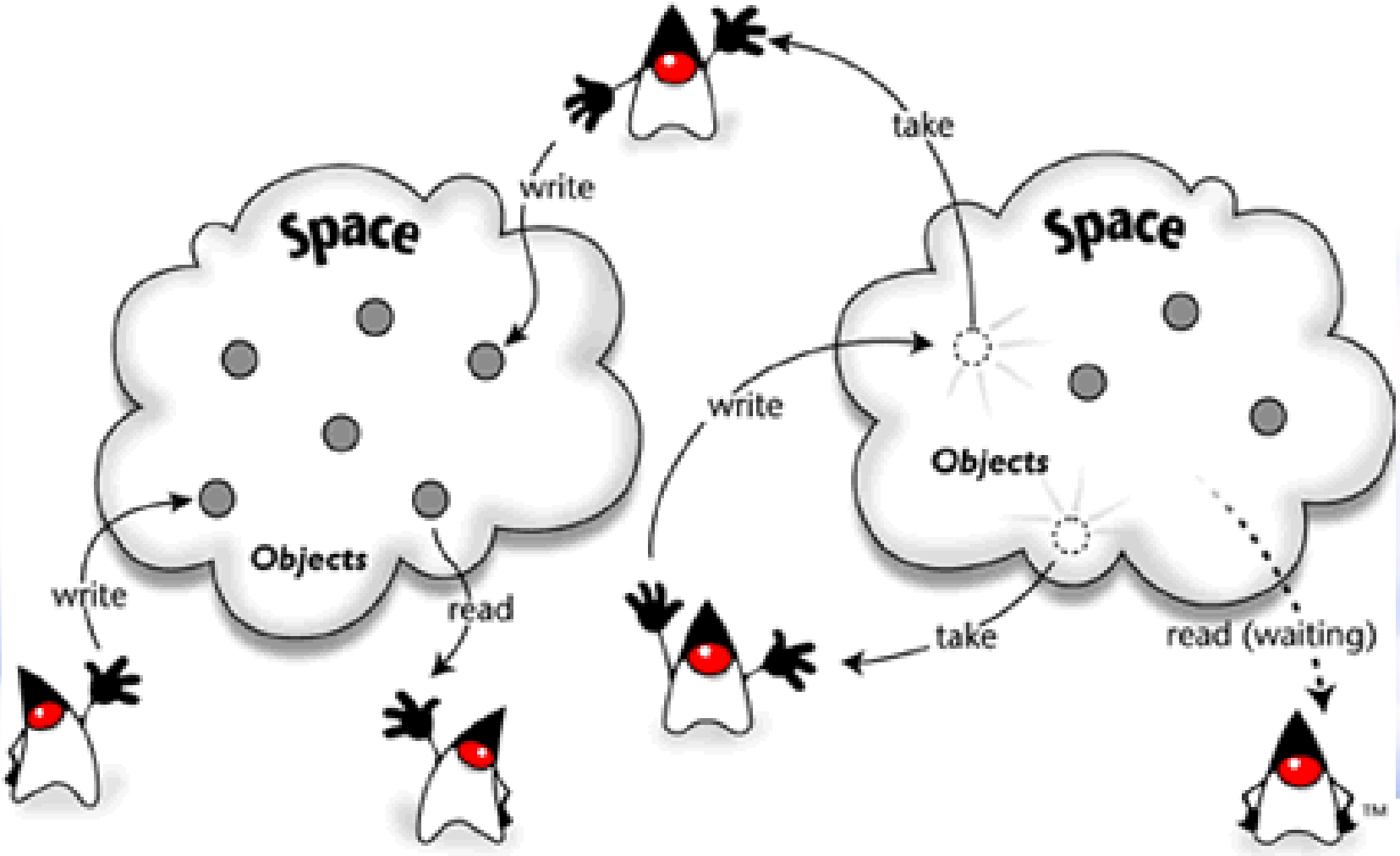
JavaSpaces

- Biblioteca oferita in cadrul Jini
- Implementare de memorii distribuite, asemanatoare cu cea din limbajul Linda
- Modelul Linda a fost completat pentru a se beneficia de tehnologia Java (RMI, serializare)
- Spatiu – utilizat in JavaSpaces pentru stocarea obiectelor in memoria distribuita
- Posibile utilizari:
  - Sisteme de chat
  - Probleme computational intensive
  - Licitatii online

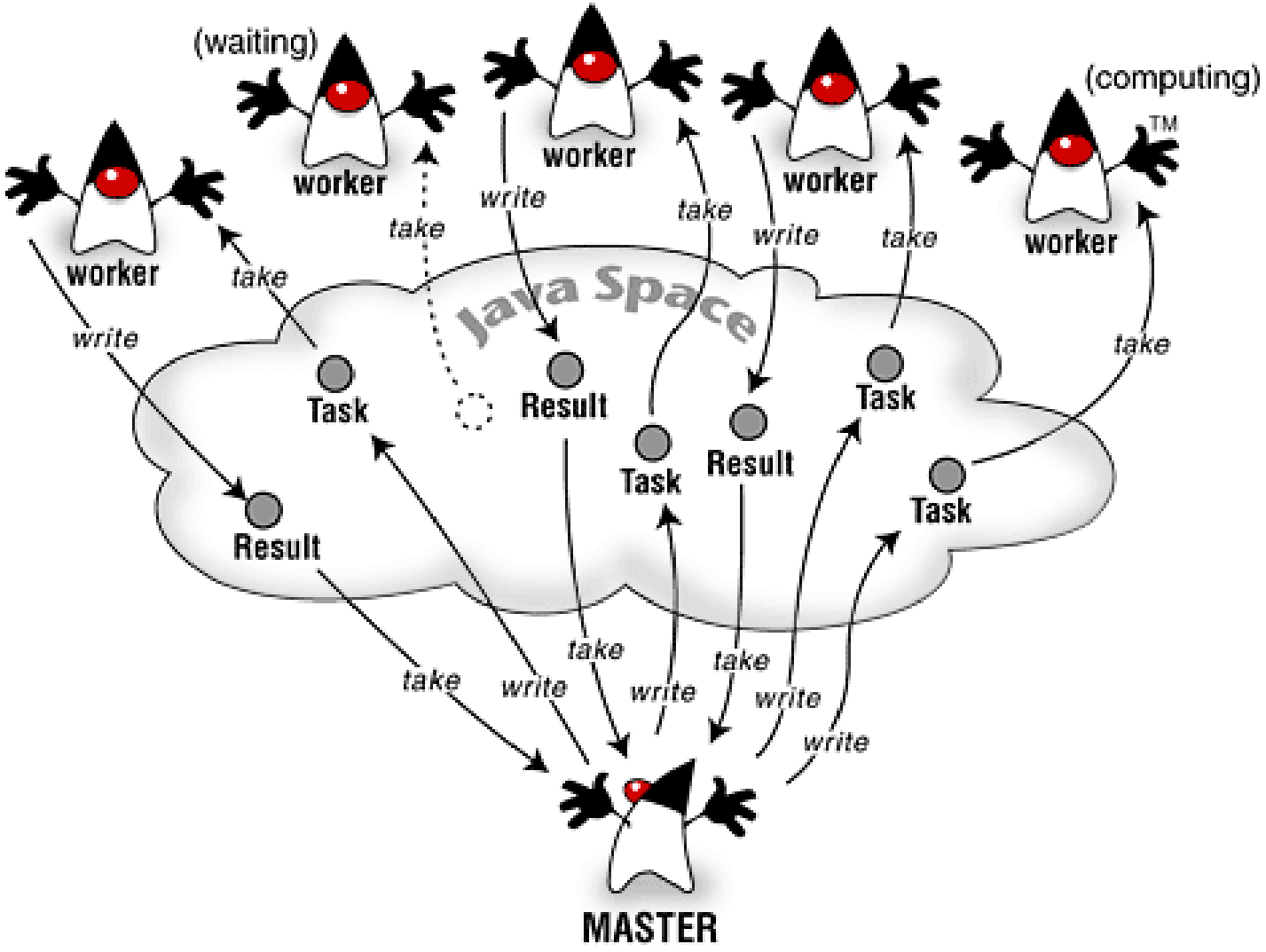
Proprietati ale spatiilor

- Partajare – pot fi accesate simultan de mai multe procese
- Persistenta – odata stocat intr-un spatiu, un obiect ramane acolo
  - pana este sters in mod explicit, sau
  - pana expira timpul de *lease*, daca exista
- Asociativitate – obiectele sunt cautate dupa continut, nu dupa locatie
- Atomicitate – operatiile sunt executate in cadrul unor tranzactii
- Pot fi utilizate pentru a transmite continut executabil (obiecte)

Interactiuni JavaSpace



Master-Worker in JavaSpace



Interfata `Entry`

- Obiectele stocate intr-un spatii JavaSpaces trebuie sa fie din clase ce implementeaza `net.jini.core.entry.Entry`
- Interfata `Entry` nu contine metode
- Alte conventii pentru obiectele stocate in spatii:
  - sa aiba un constructor public fara argumente
  - campurile sa fie publice
  - campurile sa nu contina tipuri de date primitive, ci doar referinte la obiecte
- Exemplu:

```
import net.jini.core.entry.Entry;
public class Message implements Entry {
  public String content;
  public Message() {
  }
}
```

Interfata JavaSpace

- Implementata de catre obiectele ce reprezinta spatii
- `write()` – scrierea unei intrari in spatiu
- `read()` – citirea unei intrari din spatiu care se potriveste cu modelul dat
- `take()` – extragerea unei intrari din spatiu care se potriveste cu modelul dat
- `readIfExists(), takeIfExists()`
- `notify()` – anuntarea ca a fost scrisa in JavaSpace o intrare care se potriveste cu modelul
- `snapshot()` – pentru minimizarea operatiilor de serializare

Tranzactii

- Grupuri de operatii executate in mod atomic
- Jini ofera suport pentru tranzactii: biblioteca `net.jini.transaction`
- Tranzactiile sunt obtinute de la un manager de tranzactii si pot fi utilizate pentru o perioada finita de timp
- Tranzactia este transmisa ca argument tuturor operatiilor care trebuie executate in cadrul acesteia

Tranzactii si operatiile JavaSpace

- `write()` – operatia de scriere este vizibila numai dupa ce s-a terminat tranzactia din care face parte; daca intrarea este extrasa inainte de terminarea tranzactiei, operatia nu va mai fi vizibila
- `read()` – intrarile citite in tranzactia curenta pot fi citite si din alte tranzactii, dar nu pot fi extrase
- `take()` – o intrare extrasa intr-o tranzactie nu este vizibila pentru nici o alta tranzactie
- `notify()` – o operatie `notify()` invocata intr-o tranzactie se poate referi numai la scrierile efectuate in tranzactia respectiva

Referinte

- http://www.bsdg.org/Jim/Peer2Peer/Paper/3214.html - A Guide to Peer-2-Peer
- [Ath02] Irina Athanasiu, “Java ca limbaj de programare distribuita”, editura Matrix Rom, 2002
- [Ven99] http://www.javaworld.com/jw-06-1999/jw-06-jiniology.html - “Jini: New technology for a networked world”, de B. Venners
- http://www.informit.com/articles/article.asp?p=27302&rl=1 – Introduction to JXTA
- http://www.javaworld.com/javaworld/jw-11-1999/jw-11-jiniology.html?page=1 – Make room for JavaSpaces

Referinte suplimentare

- http://compnetworking.about.com/od/p2ppeertopeer/a/p2pintroduction.htm - Introduction to P2P Software and Networking
- [Kot04] http://www.devx.com/java/Article/21947/0/page/1 - Java Dynamic Networking with Jini Technology, Part 1, de J. Kotzen
- http://www-128.ibm.com/developerworks/library/j-p2pint1.html - Making P2P Interoperable: The JXTA story

Sumar

- Sisteme Peer-to-Peer
- Jini
- Proiectul JXTA
- Comparatie JINI - JXTA
- JavaSpaces