CMSC433, Spring 2001
Programming Language
Technology and Paradigms
Jini

Alan Sussman
November 6, 2001

CMSC 433, Fall 2001 - Alan Sussman

Administrivia

• Read Chapter 15, pages 1003-1009, on Jini
• Project 4 - due tomorrow at 6PM
• 2nd midterm next Thursday, Nov. 15
  – practice exam posted by tomorrow
• Project 5 description by Thursday
• Office hours tomorrow moved to 10-11

CMCS 433, Fall 2001 - Alan Sussman

Last time

• Distributed computing issues
  – uniform view of local vs. remote objects
  – doesn’t work – need to modify interfaces, not just implementations
  • partial failures
  • true concurrency
  • latency

• Basic CORBA
  – language-independent RPC
  – standard set of software services
  • ORB, IDL, naming service

CMCS 433, Fall 2001 - Alan Sussman

Jini overview

• Framework and services on top of Java RMI
• Finding services
  – finding registries (discovery)
  – putting entries into registries (join)
  – searching registries (lookup)
  – keeping registries up to date
• Transactions
• JavaSpaces
  – Shared persistent store

CMCS 433, Fall 2001 - Alan Sussman

Providing Services

Finding providers

CMCS 433, Fall 2001 - Alan Sussman

CMSC 433, Alan Sussman, U. Maryland (via Bill Pugh)
Using Services

Proxies

- Java classes that implement standard interfaces
- Classes get downloaded as needed
  - From http server?
- Proxies can use any mechanism to communicate with actual service
  - RMI stubs, local computation and any communication medium (e.g., sockets)

Lego Mindstorm Proxies

- An example: Proxies for Lego Mindstorm robots
- Processor on Mindstorm not powerful enough to host a JVM
- Proxy can keep state about robot, communicate through infrared link

Lookup services

- Broadcast looking for lookup services messages
- Each lookup service on network responds with a proxy
- Queries to lookup service are done through proxy
  - actual communication with lookup service is up to proxy

Using a lookup service

- Can specify a list of classes/interfaces
  - only interested in results that are subtypes of all
- Can specify attributes
  - Location {
    String building;
    String floor;
    String wing;
  }
  - must match all attributes

Service ID’s

- When services first register, they are assigned a service ID
  - 128 GUID
- a particular service gets the same service ID on all lookup services
  - if you get the same service back from multiple lookup services, can detect duplicates
Leasing

- When a service registers with a lookup service, it gets a limited duration lease
  - if lease expires without being renewed, service is de-listed
- Service must renew lease
  - makes sure service is still up and still reachable from lookup service

Finding lookup services, revisited

- When a service is activated, broadcasts request to find lookup services
  - using multicast over local network
- When lookup service is activated, and at periodic intervals, broadcasts “I’m a lookup service”
  - services that learn about new lookup services register with them

Attribute lookup

- Attributes have public references for fields
  - no primitive types
- To match
  - Lookup attribute must be subtype of stored attribute
  - Each non-null field must be equal

Jini example

- Jini network among computers at home
- Jini wireless network among my cell phone, and PDA
- Don’t have to push an “I’m home” button
- Next time lookup services announce “I’m here”, devices will join

Additional Jini services

- Transactions
- JavaSpaces

Transactions

- Create a transaction from a transaction manager
  - requires maintaining a lease
- All actions performed as part of a transaction should only be visible to that transaction
  - until the transaction commits
Committing a transaction

• All participants are asked to prepare the transaction
  – log it to persistent storage
  – so that it will survive crashes
• Then commit
  – or abort

JavaSpaces

• Derived from Linda TupleSpace
• Persistent store that supports transactions
• Contains entry objects
  – i.e., records
• Stored objects are leased
  – if lease expires, entry removed

JavaSpace operations

• write - add an entry
• read - find a matching entry
  – Can wait for matching entry (with timeout)
• notify - add listening for matching entries
• take - read and delete atomically

JavaSpace matching

• Entry’s match just like Jini lookup attributes
• Can provide don’t cares
• Can use subtyping