

Lezione n.14

Laura Ricci

15/5/2009

JXTA: concetti generali

**Materiale didattico
distribuito a lezione**

Laura Ricci

JXTA: MOTIVAZIONI

- JXTA: tecnologia sviluppata da Sun Microsystems (team Bill Joy e Mike Clary) per lo sviluppo di applicazioni P2P
- JXTA: deriva da **juxtapose = giustapposto**
la tecnologia P2P si deve affiancare, piuttosto che sostituire, la tecnologia client/server (o web based)
- Motivazioni:
 - ogni applicazione P2P è basata su un **protocollo proprietario**
 - l'incompatibilità dei protocolli rende difficile l'integrazione di servizi offerti da reti P2P diverse
 - ogni rete P2P definisce una comunità chiusa, indipendente da altre comunità
 - necessità di definire un insieme di protocolli generali che possano essere utilizzati da applicazioni diverse

JXTA: SCELTE DI PROGETTO

- JXTA = insieme aperto di protocolli P2P
- Obiettivi:
 - Interoperabilità
 - Ubiquità
 - Indipendenza dalla piattaforma utilizzata

JXTA: INTEROPERABILTA'

La maggior parte dei sistemi P2P è stata progettata per applicazioni specifiche

- **Napster**: file sharing musicale
- **Gnutella**: file sharing generico
- **instant messaging**
- Ognuno di questi servizi è stato realizzato sviluppando codice specifico per quel servizio
- **Conseguenze**:
 - sistemi incompatibili, incapaci di interagire
 - replicazione dello sforzo di creare le primitive di base utilizzate da tutti i sistemi P2P utilizzati
- JXTA tenta di definire un **linguaggio comune** che possa essere utilizzato per sviluppare applicazioni diverse

JXTA: INDIPENDENZA DALLA PIATTAFORMA

La tecnologia JXTA è stata sviluppata per essere indipendente da:

- Il linguaggio di programmazione, es: esistono bindings per il Java, C, C++, C#
- Il sistema operativo, es Microsoft Windows, Linux
- Il protocollo di rete, es TCP/IP o Bluetooth

Molti client P2P offrono un'interfaccia legata ad un particolare sistema operativo ed utilizzano uno specifico protocollo di rete. Ad esempio: C++ API per Windows su TCP/IP

In JXTA l'indipendenza

- dal linguaggio è ottenuta mediante una **rappresentazione testuale dei messaggi (XML)** scambiati dai protocolli ai diversi livelli
- dal protocollo di rete utilizzato è ottenuta mediante l'introduzione di canali di comunicazione ad alto livello, **le pipes**

UBIQUITA'

Molte applicazioni P2P tendono ad essere sviluppate per un insieme particolare di dispositivi

- Problemi di portabilità
- Complessità della applicazione per poter essere eseguita su dispositivi di piccole dimensioni

JXTA: implementabile su ogni dispositivo **con un 'digital hearthbeat'**

- PDS
- Cellulari
- Sensori
- Desktop routers

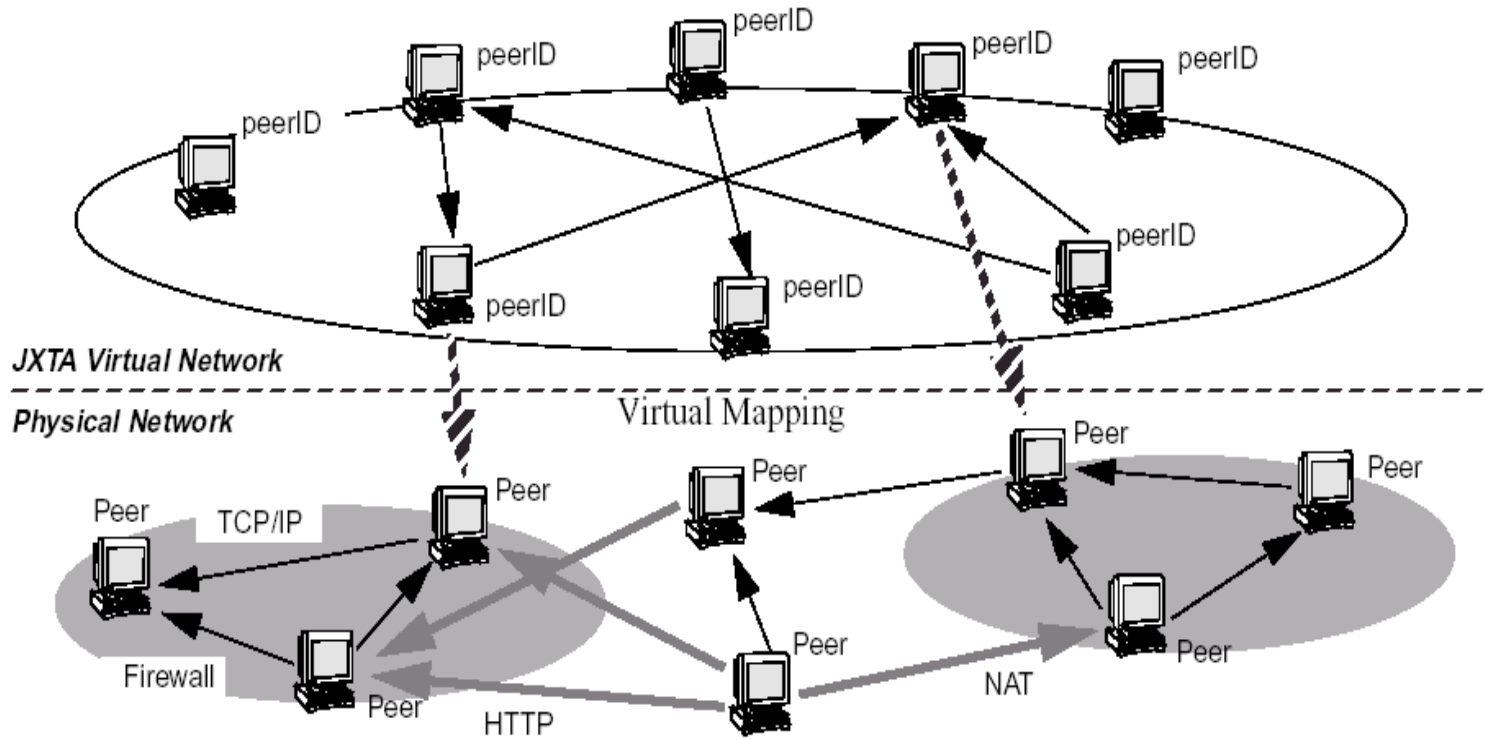
JXTA: CONCETTI GENERALI

- Elementi principali
 - Peer, Peergroups, Advertisements, Pipes, Modules, Security
- Protocolli definiti
 - Discovery Protocol
 - Pipe Binding Protocol
 - Resolver Protocol
 - Rendezvous Protocol
 - Information Protocol
 - Endpoint Protocol

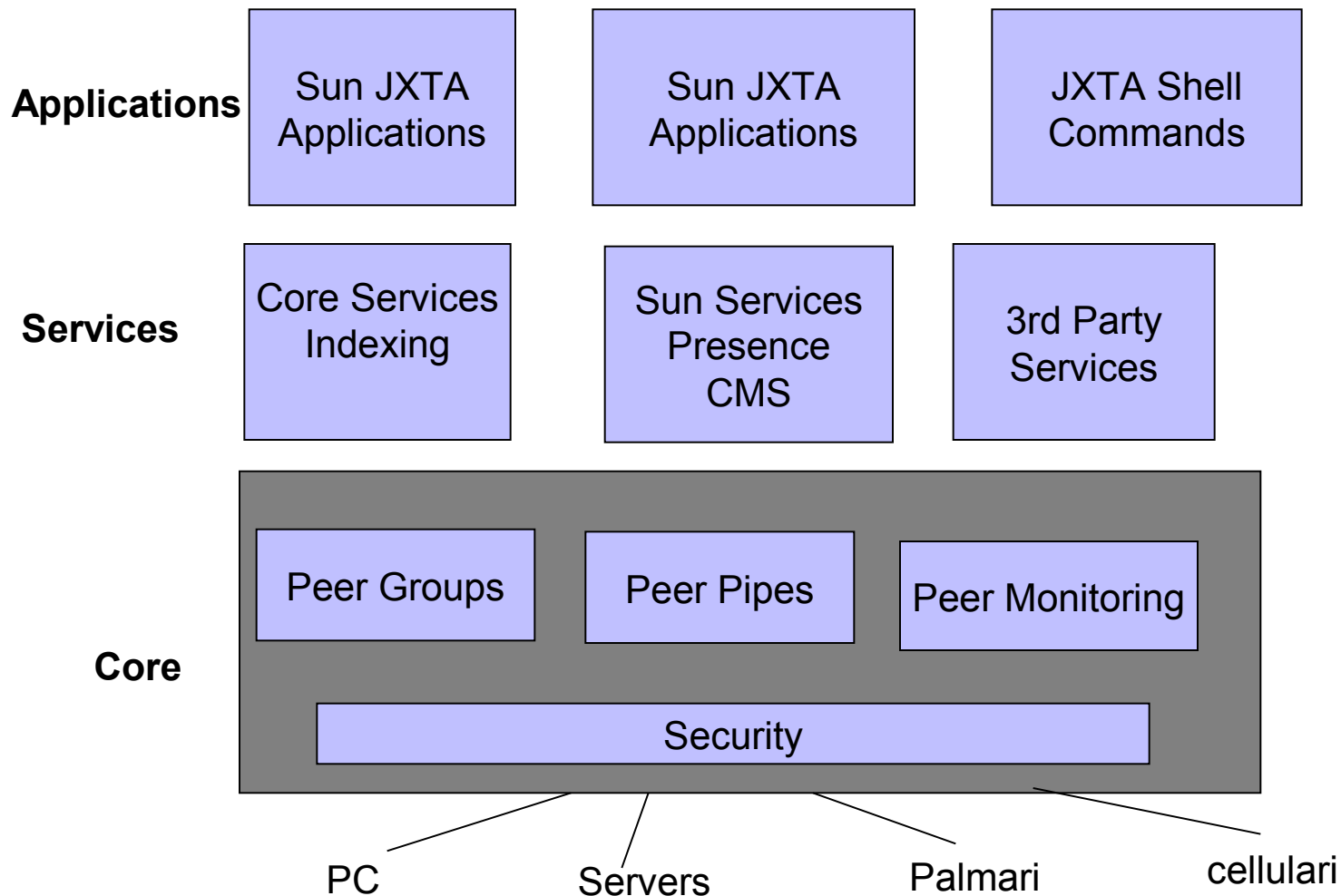
JXTA: SCELTE DI PROGETTO

- JXTA = definisce un insieme di protocolli che standardizzano il modo in cui i peer:
 - scoprono altri peer all'interno della rete
 - si auto-organizzano in peer groups
 - pubblicano e scoprono servizi di rete
 - comunicano direttamente (meccanismo delle pipes)
 - controllano l'attività degli altri peers (monitoraggio)
- ogni protocollo definisce un insieme di messaggi che vengono scambiati tra i partecipanti
- per ogni messaggio viene definito un formato pre-definito

JXTA: LA RETE VIRTUALE



JXTA LAYERS



JXTA LAYERS

- **JXTA Core:** incapsula un insieme minimo di primitive essenziali per il supporto di applicazioni P2P
 - creazione di peers e di gruppi di peers
 - sicurezza
 - scoperta di peers, servizi, risorse della rete
 - invio messaggi, gestione firewalls e NAT
- **JXTA Services:** definisce un insieme di servizi non strettamente necessari per una rete P2P, ma comuni nella maggior parte degli ambienti P2P (es: file sharing, distributed file systems,...)
- **Applicazioni:** sviluppate da SUN (MyJXTA,...) e in genere dalla comunità di utenti

JXTA: I PEERS

- **Peer**: un qualsiasi dispositivo connesso ad una rete JXTA che implementa almeno uno dei protocolli definiti da JXTA
- Ogni peer
 - è identificato da un identificatore unico (JXTA ID)
`urn:jxta:uuid-596626164626162.....COAB663666DA53903`
 - è caratterizzato da un **tipo** che definisce l'insieme di responsabilità che il peer decide di assumersi
- Classificazione dei peers
 - Peer Semplici
 - Minimal Edge Peer
 - Full Fetured Edge Peer
 - Rendezvous Peer
 - Relay (o routers) Peers

JXTA: CLASSIFICAZIONE DEI PEERS

Minimal Edge Peer

- peer con risorse limitate (palmare, cellulare,...), possibilmente allocato a monte di un NAT o di un firewall. A questi peer sono attribuite responsabilità minime
 - può inviare e ricevere messaggi
 - non inoltra messaggi e non memorizza informazioni per conto di altri peer
 - non possiede una cache per memorizzare informazioni scoperte sulla rete
-
- **Full fetured Edge Peer**
 - ha le stesse funzionalità di un minimal edge peer
 - possiede una cache locale dove memorizza informazioni ricevute dalla rete

JXTA: CLASSIFICAZIONE DEI PEERS

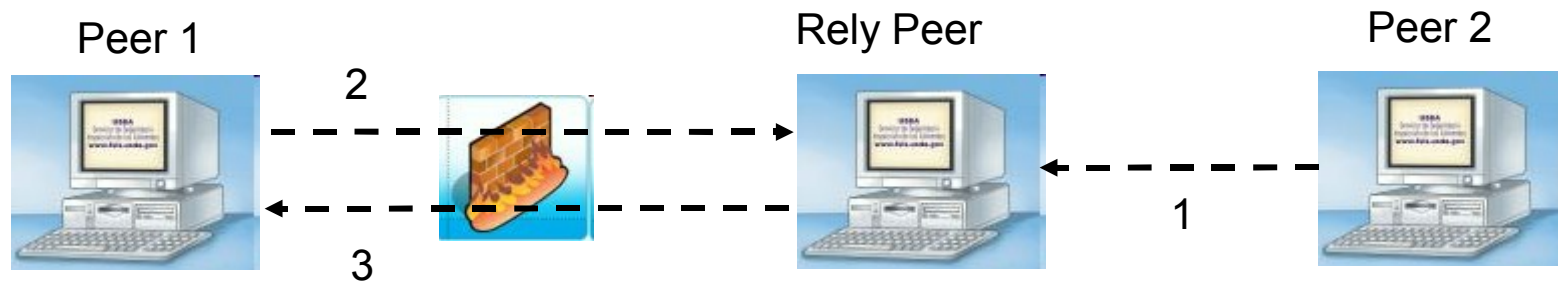
- **Rendez-vous peer:** (Rendez-vous = Punto di incontro) ha le stesse funzionalità di un full-edged peer, ma offre ulteriori servizi
 - **routing:** può inoltrare queries per conto di altri peers
 - **memorizzazione:** può memorizzare riferimenti ad informazioni memorizzate su altri peers
 - per offrire questi servizi deve possedere risorse di calcolo e di memorizzazione superiori a quelle di un edge peer
- **Relay Peer:** (Relay=fare affidamento su) può essere considerato un router JXTA per i messaggi scambiati sull'overlay JXTA
 - Proxy per un peer a monte di un firewall o di un NAT
 - Un minimal edge peer (esempio J2ME) può utilizzare un relay peer per far eseguire ad esso alcune funzioni (es: elaborazione messaggi XML,.....).

JXTA RELY PEERS

- **Rely Peers** utilizzati per permettere la connessione ad una rete JXTA di peer che si trovano all'interno:
 - di una rete protetta da un firewall oppure
 - di una rete privata i cui indirizzi privati vengono mappati da un NAT in indirizzi pubblici
- **Firewalls**
 - bloccano connessioni provenienti dall'esterno verso gli hosts della rete protetta
 - restringono le connessioni aperte verso l'esterno da hosts della rete protetta (es: solo protocollo HTTP, porte utilizzate,...)
- **NAT**
 - un peer esterno alla rete privata può comunicare con un host H della rete
 - solo se H inizia la comunicazione

JXTA RELY PEERS: FIREWALL O NAT SINGOLO

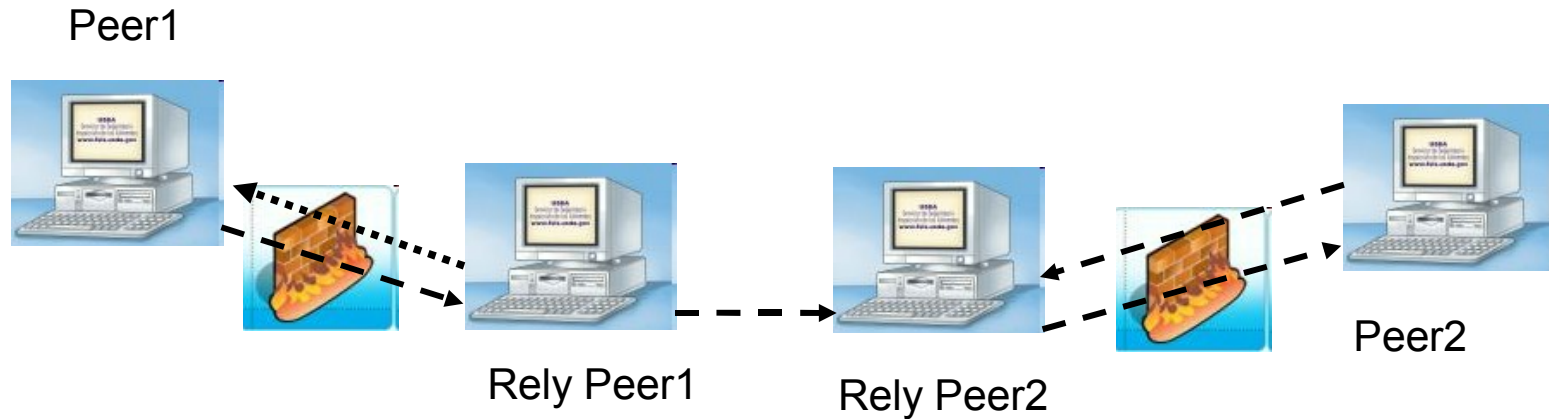
Un peer appartenente ad una rete privata protetta da un firewall utilizza un **rely peer** esterno al firewall (o appartenente alla rete privata, ma visibile dall'esterno) per attraversare il firewall



1. Il Peer2 non può connettersi con il Peer1, che si trova a monte di un firewall. Peer2 si connette al Rely Peer ed invia ad esso i messaggi destinati a Peer1. Il Rely Peer inoltra a Peer1 i messaggi ricevuti da Peer2
2. Peer1 si connette **periodicamente al rely peer** per dare ad esso la possibilità di inoltrare i messaggi ricevuti da Peer2. Peer1 utilizza un protocollo che gli consente di attraversare il firewall (es: HTTP)
3. Il rely peer invia a Peer1, per conto di Peer2 i messaggi ricevuti

JXTA RELY PEERS: FIREWALL/NAT MULTIPLI

Se i due peer che intendono comunicare si trovano entrambi a monte di un firewall/NAT si utilizza una coppia di rely peers



JXTA RELY PEERS

- Per inviare un messaggio ad un peer posizionato a monte di un firewall, occorre conoscere il rely peer corrispondente
- Questa informazione può essere ottenuta mediante un processo di discovery di routing informations effettuato in modo distribuito
- L'invio di messaggi diversi tra la stessa coppia di peers può utilizzare diversi rely peers

JXTA: PEER GROUPS

- nelle applicazioni P2P 'classiche' (Gnutella, ICQ, Freenet), ogni applicazione definisce una diversa overlay network ed ogni overlay network utilizza un protocollo diverso
- in JXTA: lo stesso insieme di protocolli utilizzato per applicazioni diverse
- **JXTA Peer Groups:** suddividono l'overlay network JXTA in base dell'applicazione a cui i peer partecipano
- **Peer Group=** insieme di peers caratterizzati da **interessi o scopi comuni**. Ogni gruppo fornisce ai partecipanti un insieme di servizi che non sono accessibili da peers JXTA che non partecipano al gruppo
- Esempio: file sharing peer group, CPU sharing peer group

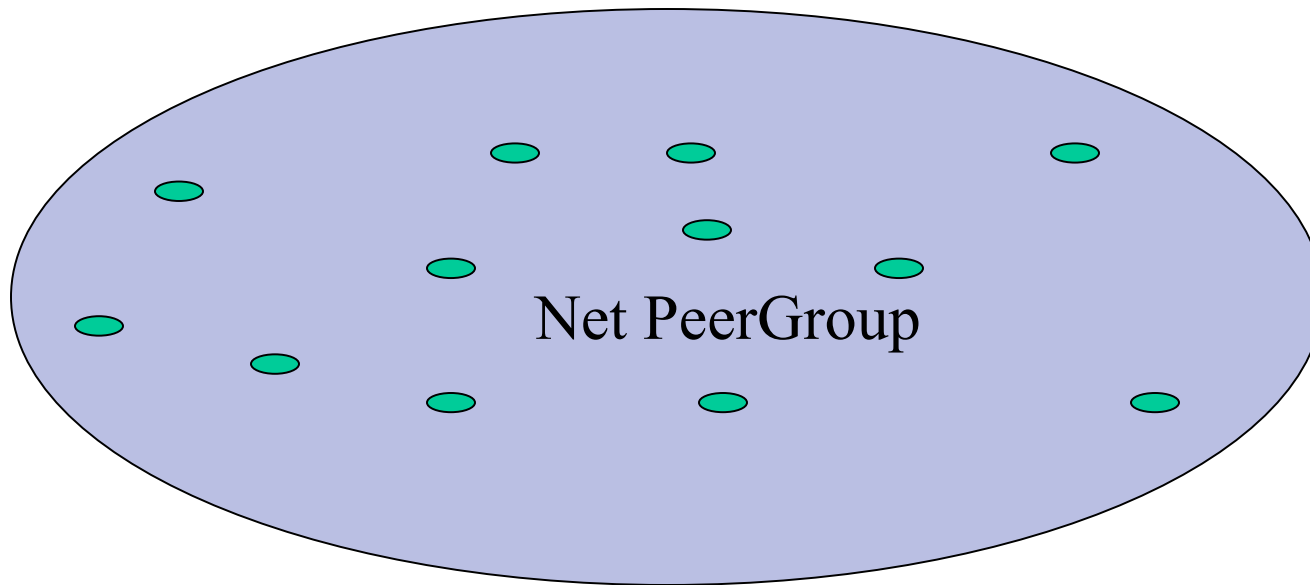
JXTA: PEER GROUPS

- JXTA definisce un insieme di protocolli per gestire la creazione dei gruppi, la richiesta di partecipazione ad un gruppo,...
- I gruppi vengono creati dalle applicazioni
- I peer groups definiscono delle 'regioni astratte' all'interno della rete JXTA
- Un peer group viene utilizzato per definire:
 - un insieme di **servizi** e risorse accessibili dai partecipanti al gruppo
 - un insieme di **politiche di sicurezza** associate al gruppo (membership requirements)
 - un meccanismo di **scoping**
 - un ambiente di **monitoraggio**

JXTA: PEER GROUPS

- i peer si organizzano dinamicamente in peer groups
- all'inizio della esecuzione ogni peer appartiene, per default, a `NetPeerGroup`, un gruppo che fornisce un insieme di servizi base
- ogni peer può unirsi a diversi gruppi durante la sua esecuzione e può partecipare a più peergroups simultaneamente
- ogni gruppo viene creato a partire da un un unico gruppo esistente
- `Albero di peer groups` = descrive le relazioni tra i gruppi attivi durante l'esecuzione di un programma JXTA
- Ogni gruppo offre un insieme di servizi ai peer che partecipano al gruppo

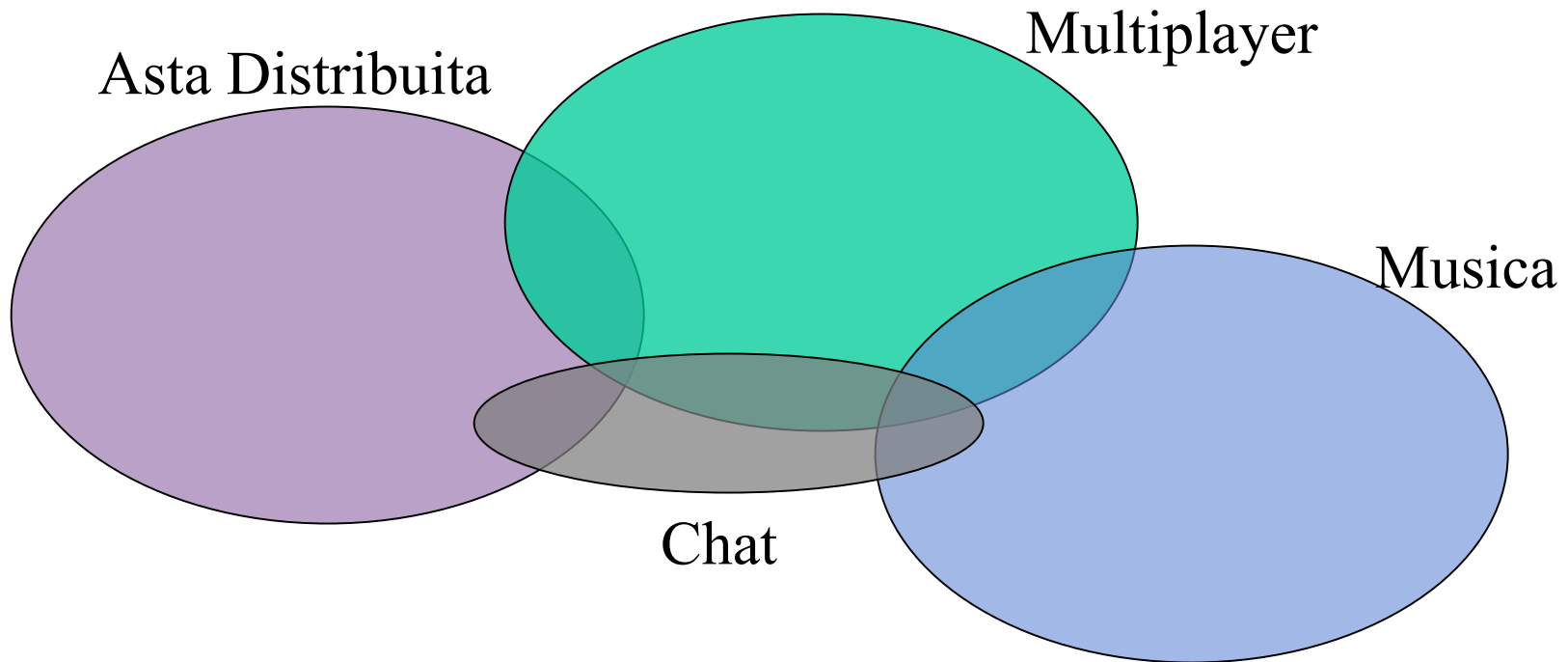
IL NETPEERGROUP



All'inizio della esecuzione

- esiste un peer group creato di default, il [NetPeerGroup](#)
- ogni peer è membro del NetPeerGroup
- può scoprire risorse all'interno di quel peer group e comunicare con altri peer appartenenti ad esso

JXTA PEER GROUPS



- Ogni peer può **creare**, **unirsi** e **lasciare** un PeerGroup
- Ogni peer può appartenere a più di un peer group contemporaneamente
- Le ricerche di risorse vengono effettuate rispetto ad un peergroup di riferimento

JXTA: ADVERTISEMENTS

- **Advertisement** = Rappresentazione strutturata di una **entità**, di un **servizio**, o di una **risorsa** resa disponibile da un peer all'interno di un peer group
- Rappresentazione mediante documenti XML
- Ogni peer
 - **pubblica** un insieme di advertisements per descrivere le risorse che mette a disposizione
 - **ricerca** advertisements sulla rete
- Ogni advertisement è caratterizzato da un **ciclo di vita**, dopo cui viene eliminato dalla rete
- Ogni peer mantiene una **cache** locale in cui memorizza gli advertisements ricevuti dalla rete

JXTA: UN PEER ADVERTISEMENT

```
<?xml version="1.0"?>
<!DOCTYPE jxta:PA>
<jxta:PA xmlns:jxta="http://jxta.org">
  <PID>
    urn:jxta:uuid-
59616261646162614A78746150325033F3BC76FF13C2414CBC0AB663666DA53903
  </PID>
  <GID>
    urn:jxta:jxta-NetGroup
  </GID>
  <Name>
    suz
  </Name>
  <Svc>
    <MCID>
      urn:jxta:uuid-DEADBEEFDEAFBABAFAFEEDBABE0000000805
    </MCID>
    <Parm>
      <Addr>
        tcp://192.168.1.100:9701/
      </Addr>
      <Addr>
        jxtatls://uuid-
59616261646162614A78746150325033F3BC76FF13C2414CBC0AB663666DA53903/
TlsTransport/jxta-WorldGroup
      </Addr>
      <Addr>
        jxta://uuid-
59616261646162614A78746150325033F3BC76FF13C2414CBC0AB663666DA53903/
      </Addr>
      <Addr>
        http://JxtaHttpClientuuid-
59616261646162614A78746150325033F3BC76FF13C2414CBC0AB663666DA53903/
      </Addr>
    </Parm>
  </Svc>
</jxta:PA>
```

JXTA: TIPI DI ADVERTISEMENTS

- peer advertisements
 - nome del peer, identificatore del peer, endpoints, se è un rendez-vous peer,...
- peer group advertisement
 - nome del gruppo, identificatore, servizi offerti,....
- pipe advertisement
 - anche una pipe è considerata una risorsa condivisa. Il peer che la crea, *ne notifica l'esistenza* agli altri peer del gruppo tramite un advertisement
 - identificatore, tipo della pipe,....
- peer info advertisements
 - contiene informazioni sullo stato del peer, ad esempio, l'*uptime*, quanti messaggi ha ricevuto in un certo intervallo di tempo.
- rendezvous Advertisements
- module Specification/Implementation Advertisements

Un peer P può creare specifici tipi di advertisements per descrivere risorse/ servizi pubblicati da P stesso

CM: CACHE MANAGEMENT IN JXTA

JXTA cache management system:

- è utilizzato per memorizzare informazioni relative alla rete P2P a cui ogni peer partecipa
- contiene principalmente gli advertisements creati dal peer oppure quelli reperiti dal peer all'interno della rete
- semplice gestione degli advertisements (nelle prime versioni un file per ogni advertisement, il nome del file coincide con l'ID dell'advertisement)
- Vengono create **directory diverse** per memorizzare gli advertisements di ogni gruppo a cui partecipo
- L'introduzione della cache è motivata dall'alto costo del reperimento delle informazioni sulle risorse distribuite su una rete che contiene centinaia di peers

CM: CACHE MANAGEMENT IN JXTA

Esempio: per individuare un peer con cui voglio chattare devo reperire l'advertisement che lo descrive. Il reperimento dell'advertisement può richiedere l'attraversamento di diversi rendez-vous peers

- Se dovessi ripetere la ricerca ogni volta che voglio contattare il peer remoto, la performance dell'applicazione verrebbe ridotta notevolmente
- Il traffico sulla overlay network crescerebbe inoltre in modo esponenziale

Struttura della cache :

- **CM (Cache Management Directory)** : crea la prima volta che il peer viene eseguito, è la root directory
- **Sottodirectories:** ne esiste una per ogni gruppo a cui il peer partecipa
- La directory per il NetPeer Group esiste di default
- Le directory associate ad ogni gruppo contengono gli advertisements reperiti/pubblicati in quel gruppo ed anche altre informazioni

ADVERTISEMENT: PUBBLICAZIONE

- **Pubblicazione di un advertisement:** memorizza l'advertisement nella cache locale
- Dopo che l'advertisement è stato pubblicato, è possibile reperirlo mediante il discovery protocol
- Per accelerare la distribuzione degli advertisements nella rete JXTA fornisce metodi per la **pubblicazione remota** di un advertisement
 - invia l'advertisement a tutti gli altri peers dello stesso gruppo
 - indica l'intervallo di tempo in cui un advertisement rimane valido nelle caches dei peer remoti

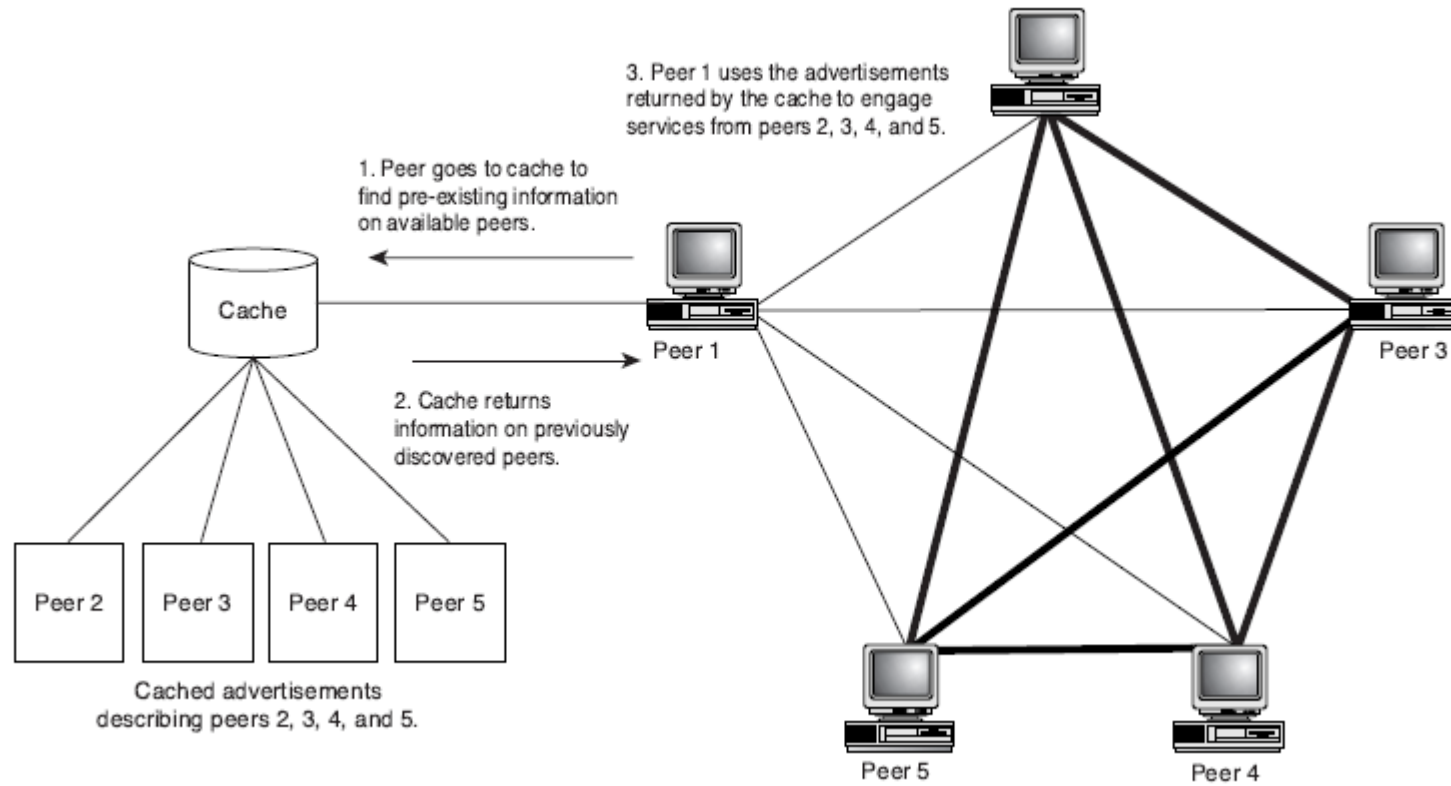
ADVERTISEMENTS: RICERCA LOCALE E REMOTA

- **Discovery Service:** utilizza il Discovery Protocol per reperire gli advertisements
- **Local Discovery:**
 - la ricerca dell'advertisement avviene nella cache locale
 - l'advertisement deve essere stato precedentemente ricercato sulla rete e caricato nella cache locale
 - prima di effettuare una ricerca remota è sempre meglio controllare se l'advertisement esiste nella cache locale
- **Remote Discovery**
 - ricerca l'advertisement nella cache di uno o più peers remoti
 - utilizzato per popolare periodicamente la cache locale

ADVERTISEMENTS: RICERCA LOCALE E REMOTA

- Le primitive di ricerca remota degli advertisements sono **non bloccanti**
- La ricerca remota **avvia un thread** che **invia una query sulla rete** per la ricerca degli advertisements, quando il thread è stato avviato, il programma continua la propria esecuzione
- Quando gli advertisements sono stati reperiti, vengono memorizzati nella cache locale del peer in modo da essere successivamente reperiti dal programma mediante una local discovery
- Questa procedura può essere utilizzata per popolare periodicamente la cache locale
- Il problema di questo approccio è che il peer non ha uno strumento per capire quando un advertisement è stato caricato nella cache locale
- Approccio alternativo: utilizzo di **listener associati al processo di discovery**, che siano in grado di catturare l'evento 'caricamento di advertisement nella cache' e di avvertire l'applicazione

RICERCA LOCALE DI ADVERTISEMENTS



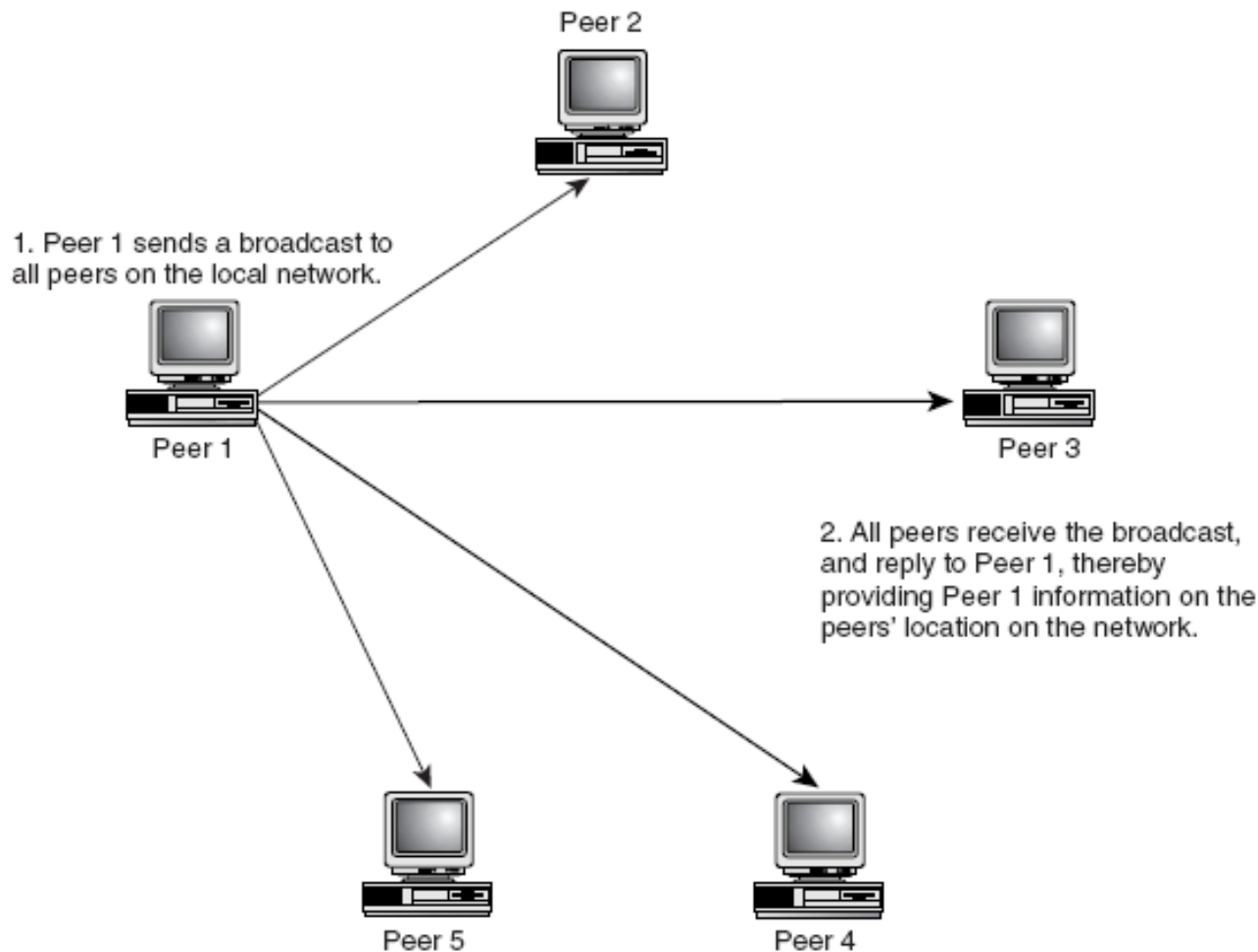
RICERCA LOCALE DI ADVERTISEMENTS

- La cache locale può contenere solo informazioni minimali (esempio indirizzi IP e porte di peers scoperti precedentemente) oppure può rappresentare un vero e proprio database di informazioni scoperte anche in sessioni precedenti
- Ricerca locale:
 - Vantaggio: elimina il costo della ricerca remota
 - Svantaggio: le informazioni reperite possono **essere datate** e descrivere risorse che non sono più disponibili sulla rete
 - Se un peer utilizza un advertisement "scaduto" il traffico sulla rete in realtà aumenta, perchè esso dovrà rendersi conto che la risorsa non esiste e quindi ricercarla di nuovo
 - Utilizzo di **expiration times**

CICLO DI VITA DI UN ADVERTISEMENT

- Ogni advertisement è caratterizzato da un suo ciclo di vita
- **Expiration Time:** esiste un valore di default, ma può essere impostato anche dal programmatore JXTA
- Ogni peer può ripubblicare regolarmente un advertisement, in modo da produrne una versione aggiornata dopo il suo expiration time
- E' possibile rimuovere advertisements dalla local cache (flushing)

RICERCA DI ADVERTISEMENTS IN RETE LOCALE



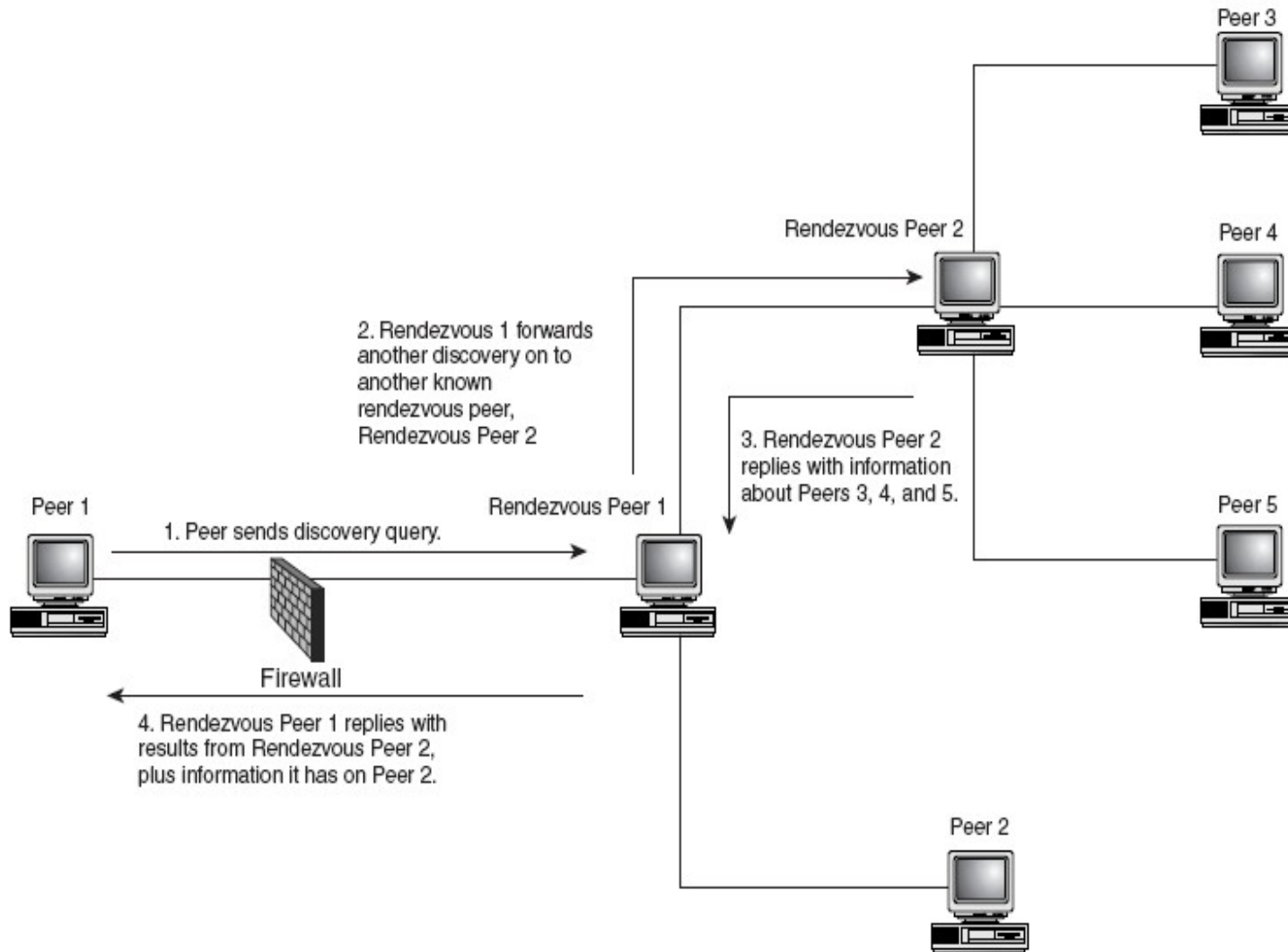
RICERCA DI ADVERTISEMENTS IN RETE LOCALE

- Se l'applicazione viene eseguita in LAN, si utilizza un **meccanismo diretto** di ricerca degli advertisements
- Utilizzato il meccanismo di **broadcast** o di **multicast** offerto dai livelli di rete inferiori
- La richiesta di discovery di adv viene inviata a tutti i peer della LAN in multicast.
- Se non c'è nessun peer rendezvous la richiesta di discovery non "esce" dalla LAN perchè nessuno la propaga fuori; quindi la ricerca di risorse viene eseguita solo nella LAN.
- Se invece c'è un rendezvous peer sarà questo a inviare la richiesta a tutti i peer che conosce (e che potrebbero essere fuori dalla LAN).

RICERCA DI ADVERTISEMENTS SU WAN

- Se un peer P è collegato ad una WAN, la ricerca degli advertisements può avvenire solo mediante un **rendez vous peer**
- Un rendez vous peer risponde alla richiesta di ricerca del peer P può
 - Propagare la richiesta ad altri rendezvous peers
 - Usare gli advertisements mantenuti nella propria cache
- Ogni peer deve connettersi a uno o più rendezvous peers
- Il meccanismo dei rendezvous peer può essere utilizzato anche in rete locale

RICERCA DI ADVERTISEMENTS SU WAN



PIPES: MECCANISMI DI COMUNICAZIONE

- Meccanismi definiti da JXTA per realizzare lo scambio di messaggi tra servizi/ applicazioni
- **Pipe**: canale di comunicazione virtuale tra due peers (astrae il concetto di socket)
- **Pipe Binding Protocol** : virtualizza il canale di comunicazione tra due peers
Esempio: la connessione tra i due peer può utilizzare più protocolli in modo trasparente rispetto all'utente
- Ad esempio, può essere utilizzato il protocollo HTTP per raggiungere un client allocato a monte di un firewall
- Il meccanismo delle pipe definisce un'astrazione che fornisce l'illusione di disporre di **mailboxes virtuali** non fisicamente collegate ad uno specifico peer.

JXTA PIPES

- Pipes canali unidirezionali ed asincroni.
 - unidirezionali: se i livelli di rete utilizzano HTTP, la comunicazione bidirezionale non è facilmente implementata
 - asincroni: non esiste alcuna sincronizzazione tra mittente e destinatario al momento dell'invio del messaggio
- Definiti diversi tipi di pipes (unicast, propagate, secure, bidipipes)
- Pipe EndPoint: estremo della pipe
 - input pipe (estremo da cui si ricevono i dati)
 - output pipe (estremo su cui si immettono i dati)
- Peer EndPoint: astrae il concetto di interfaccia di un peer (Es: indirizzo IP+porta), i pipes endpoint vengono collegati ai peer endpoints

JXTA PIPES

- Le pipes sono **risorse condivise** all'interno di un peer group
- Ogni pipe è descritta mediante un pipe advertisement
- Uno dei peer del gruppo
 - Crea la pipe, ovvero **crea un advertisement** per quella pipe e lo pubblica all'interno del gruppo
 - Gli altri peer possono ricercare questo advertisement per reperire la pipe ed utilizzarla
- Per comunicare, una coppia di peer deve:
 - utilizzare la stessa pipe
 - il destinatario connette dinamicamente un estremo della pipe, l'**input pipe** ad un suo endpoint
 - il mittente connette dinamicamente l'altro estremo della pipe, l'**output pipe**, ad un suo endpoint (output pipe)

JXTA PIPES

- Il processo di creazione delle pipes deve avvenire secondo un ordine preciso, che, in generale è il seguente
 - il destinatario
 - crea l'advertisement della pipe
 - crea l'input pipe
 - Il mittente
 - ricerca l'advertisement della pipe
 - il mittente crea l'output pipe
- Sia la creazione della input pipe che quella della output pipe avviene a partire da un pipe advertisement

JXTA: UN PIPE ADVERTISEMENTS

```
<?xml version="1.0"?>
```

```
<!DOCTYPE jxta: PipeAdvertisement>
```

```
<jxta: PipeAdvertisement xmlns:jxta="http://jxta.org">
```

```
  <id>
```

```
urn:jxta:uuid
```

```
59616261646162614E504720503250338E3E786229EA460DADC1A176B69B7354
```

```
  </id>
```

```
  <type>
```

```
    JxtaUnicast
```

```
  </type>
```

```
  < name
```

```
    TestPipe
```

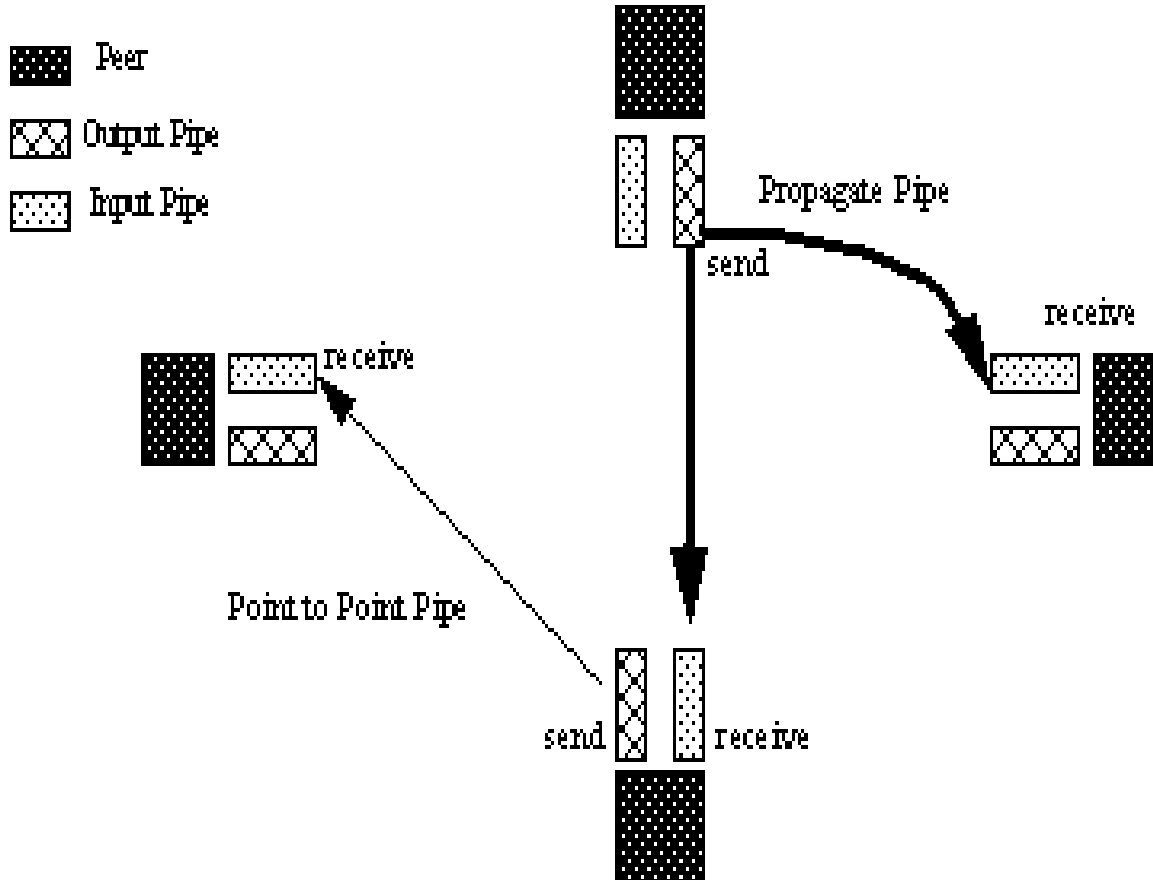
```
  </name>
```

```
</jxta:PipeAdvertisement>
```

JXTA: TIPI DI PIPES

- **Point-to-point (Unicast) Pipe:** mette in collegamento unidirezionale tra due peer.
- **Secure Unicast Pipes:** point to point pipe che fornisce un servizio di comunicazione
- **Propagate Pipes:** pipe di tipo **uno a molti**. Connette una output pipe ad un insieme di input pipes. Le input e le output pipes devono appartenere al solito gruppo
 - Esempio: implementazione di un servizio di chat. Un messaggio deve essere trasmesso a tutti i partecipanti ad una chat
- **BidiPipes:** bidirectional pipes: wrapper che utilizza due pipes unidirezionali in cui i dati fluiscono in direzioni opposte

JXTA: TIPI DI PIPES



JXTA PIPES

- Gli estremi di una pipe vengono collegati dinamicamente ad uno o più peer endpoints, mediante il pipe binding protocol.
- **Blind pipes**; il mittente può utilizzare una pipe senza conoscere il peer a cui è collegata la output pipe
- Nelle applicazioni P2P spesso non è tanto importante conoscere il peer a cui ci si connette, quanto il tipo di risorse che il peer offre

Esempio: un peer pubblica una pipe per offrire un certo servizio

- la stessa pipe può quindi essere collegata a tempo di esecuzione a diversi peer
 - un peer guasto P1 può essere sostituito da un nuovo peer P2
 - P2 prende la responsabilità di gestire le comunicazioni dirette a P1, collegando ad un proprio end point gli estremi delle pipe utilizzate in precedenza da P1