

Università degli Studi di Padova

Gestione dei nomi



Anno accademico 2004/5
Corso di Sistemi Concorrenti e Distribuiti

Tullio Vardanega, tullio.vardanega@math.unipd.it

Corso di Laurea Specialistica in Informatica, Università di Padova 1/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Denominazione di entità - 1

- ❑ Un nome in un sistema distribuito è una stringa (di *bit* or di caratteri) usata come denotazione di entità
- ❑ Le entità necessitano di denotazioni che le rendano utilizzabili
- ❑ Le entità hanno punti di accesso (*access point*) il cui nome è l'indirizzo dell'entità
- ❑ Una entità può avere più punti di accesso
 - Più punti di vista simultanei o diversi nel tempo

Corso di Laurea Specialistica in Informatica, Università di Padova 2/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Denominazione di entità - 2

- ❑ **Trattare gli indirizzi come nomi speciali**
 - Consente di decomporre la corrispondenza entità-indirizzo in 2 relazioni legate ma distinte
 - Nome di entità → nomi degli attributi dell'entità (tra cui punto di accesso)
 - Punto di accesso → indirizzo dell'entità
 - Un'entità può cambiare punto di accesso
 - Un punto di accesso può essere riassegnato da un'entità all'altra
 - Un'entità può avere più punti di accesso
- ❑ **Vogliamo nomi indipendenti dagli indirizzi (*location independence*)**

Corso di Laurea Specialistica in Informatica, Università di Padova 3/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Denominazione di entità - 3

- ❑ È desiderabile che certi nomi possano essere utilizzati per designare entità univocamente
 - **Identificatori**
 - Non tutti i nomi sono identificatori!
 - Alcuni nomi devono essere "trattabili" (*human-friendly*), ma non gli indirizzi né gli identificatori
- ❑ Un vero identificatore denota una entità o nessuna, non è riusabile, e l'entità non può averne più di uno
 - In questo caso, identificatori diversi designano entità diverse
- ❑ Nomi diversi per una stessa entità sono detti alias

Corso di Laurea Specialistica in Informatica, Università di Padova 4/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Denominazione di entità - 4

- ❑ **Lo spazio dei nomi (*name space*)** 1/2
 - Rappresentabile come un grafo diretto etichettato, con 2 tipi di nodo
 - **Nodo terminale** (foglia) → da cui non diparte alcun arco, contiene informazioni sull'entità che esso rappresenta (p.es. l'indirizzo, lo stato)
 - **Nodo repertorio** (*directory*) → da cui dipartono archi etichettati con un nome, contiene una tabella di corrispondenze <etichetta d'arco, identificatore di nodo>
 - Il nodo con solo archi in uscita è detto nodo radice (di un *name space*)
 - Uno spazio dei nomi può ammettere più nodi radice
 - Ogni nodo è un'entità del sistema distribuito ed è associato ad un identificatore

Corso di Laurea Specialistica in Informatica, Università di Padova 5/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Denominazione di entità - 5

- ❑ **Lo spazio dei nomi** 2/2
 - Ogni cammino sul grafo (*path name*) è denotato dalle etichette degli archi percorsi
 - Cammino assoluto, se il nodo di origine è la radice del grafo
 - Cammino relativo, altrimenti
 - Un nome è sempre definito in relazione ad un nodo repertorio
 - Nome globale, se viene sempre interpretato rispetto allo stesso nodo repertorio
 - Nome locale, se la sua interpretazione dipende dal contesto d'uso
 - Il grafo dei nomi è spesso aciclico, ma non tutti lo sono

Corso di Laurea Specialistica in Informatica, Università di Padova 6/34

Sistemi distribuiti: gestione dei nomi

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Denominazione di entità - 6

□ **Risoluzione dei nomi**

- **Look-up:** restituisce l'identificatore del nodo da cui proseguire il processo di risoluzione
 - **Esempio:** nel grafo dei nomi di un *file system* in UNIX e Linux l'identificatore di un nodo è il numero di indice di un particolare *i-node*; questo ci dice l'indirizzo dei dati su disco; i dati infine ci forniscono il *name space* dove proseguire la risoluzione
- **Closure:** determina il nodo dal quale il processo di risoluzione ha inizio
 - **Esempio:** nel grafo dei nomi di un *file system* in UNIX e Linux il nodo radice è sempre il primo *i-node* della partizione che rappresenta il *file system*
- **Alias**
 - **Hard link:** più cammini assoluti denotano lo stesso nodo di un grafo dei nomi
 - **Symbolic link:** il nodo terminale contiene, come attributo, il cammino assoluto

Corso di Laurea Specialistica in Informatica, Università di Padova 7/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Denominazione di entità - 7

Questo grafo dei nomi ha un nodo radice unico, di nome implicito

Leaf node ○
Directory node □

Corso di Laurea Specialistica in Informatica, Università di Padova 8/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Denominazione di entità - 8

□ **La risoluzione dei nomi può avvenire su più spazi dei nomi uniti trasparentemente**

- **Il principio del mount su un *file system*, tramite il quale un nodo repertorio di un grafo dei nomi diventa la radice di uno spazio dei nomi esterno importato**
- **Per accedere allo spazio dei nomi esterno occorre sapere**
 - Il nome del protocollo di accesso al nodo che lo ospita
 - Il nome del server che lo gestisce
 - Il nome della radice designata in esso

Corso di Laurea Specialistica in Informatica, Università di Padova 9/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Denominazione di entità - 9

□ **La realizzazione di uno spazio dei nomi può avere 3 livelli gerarchici**

- **Livello globale** → comprende i nodi di più alto livello
 - Radici e rispettivi figli (informazione generalmente stabile)
- **Livello amministrativo** → raggruppa i nodi repertorio che sono amministrati da una singola responsabilità
 - Ciascun nodo repertorio rappresenta gruppi di entità appartenenti alla stessa organizzazione od unità amministrativa
- **Livello gestionale** → consiste dei nodi che cambiano frequentemente
 - Ciascun nodo viene gestito in cooperazione dall'utente e dall'amministratore di sistema

Corso di Laurea Specialistica in Informatica, Università di Padova 10/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Denominazione di entità - 10

I 3 livelli nello spazio dei nomi realizzato utilizzando il DNS di Internet

Corso di Laurea Specialistica in Informatica, Università di Padova 11/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Denominazione di entità - 11

□ **Name resolver**

- L'entità localmente responsabile per la risoluzione dei nomi

□ **Name server**

- L'entità che gestisce i nomi del proprio repertorio

□ **2 tecniche di risoluzione dei nomi**

- **Iterativa** → ogni *name server* interrogato dal *resolver* fornisce la migliore risposta in suo possesso, senza fare ricerca
 - Il *resolver* determina il nome (cammino) completo dell'entità desiderata e lo fornisce al *name server* radice
 - Il cliente svolge più lavoro del server → non conviene fare *caching* presso il cliente
- **Ricorsiva** → ogni *name server* interrogato interroga a sua volta i *name server* di livello inferiore per costruire la risposta richiesta
 - Il *name server* radice fornisce al *resolver* il nome (cammino) completo dell'entità richiesta
 - Il server svolge più lavoro del cliente → conviene fare *caching* presso il cliente

Corso di Laurea Specialistica in Informatica, Università di Padova 12/34

Sistemi distribuiti: gestione dei nomi

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 1

- ❑ I nomi dei livelli globale ed amministrativo cambiano di rado → il contenuto dei nodi dei loro grafi è stabile
 - Il *caching* delle relative risoluzioni è conveniente
- ❑ I nomi del livello gestionale cambiano più frequentemente
 - *Caching* non conveniente → occorre minimizzare i costi di aggiornamento (del contenuto dei nodi) e di *look-up*

Corso di Laurea Specialistica in Informatica, Università di Padova 13/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 2

- ❑ Non è bene cambiare i nomi usati come identificatori, altrimenti si invalidano i *symbolic link* che li utilizzano
- ❑ Modificare lo spazio dei nomi del nodo repertorio di origine associando
 - Il nuovo indirizzo al vecchio nome → *look-up* veloce, ma ogni successivo aggiornamento (su nodo di origine) ha costo molto elevato
 - Il nuovo nome al valore del vecchio nome (= *symbolic link*) → *look-up* più lento, ma facile aggiornamento tramite nuovo *symbolic link* locale, al costo di un'ulteriore indizione

Corso di Laurea Specialistica in Informatica, Università di Padova 14/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 3

- ❑ Entrambe le soluzioni sono inadeguate per sistemi distribuiti a larga scala
- ❑ Serve una tecnica che separi i nomi dagli indirizzi (non come nel DNS di *Internet*)
- ❑ Gli identificatori sono adatti a questo scopo
 - Da nome utente ad identificatore → *naming service*
 - Da identificatore ad indirizzo → *location service*

Corso di Laurea Specialistica in Informatica, Università di Padova 15/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 4

Corrispondenza *diretta* (1 livello) tra nomi ed indirizzi → come nel DNS di *Internet*

Corrispondenza *indiretta* (2 livelli) con uso di identificatori come tramite di risoluzione

Corso di Laurea Specialistica in Informatica, Università di Padova 16/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 5

- ❑ Un *location service* accetta un identificatore di entità e ne restituisce l'indirizzo corrente
 - Un indirizzo per ogni copia (o *access point*) dell'entità
- ❑ Soluzione brutale: *broadcasting*
 - ARP (*Address Resolution Protocol*) restituisce l'indirizzo di livello collegamento dati (rete locale) corrispondente ad un indirizzo IP (*Internet*)
 - Troppo oneroso per reti vaste → *multicasting* verso gruppi specifici (anche dinamici) di nodi coinvolti

Corso di Laurea Specialistica in Informatica, Università di Padova 17/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 6

- ❑ Soluzione meno brutale: *forwarding*
 - Simile ad una catena di *symbolic link* da ogni locazione precedente verso la successiva
 - 3 importanti difetti
 - Il costo di localizzazione di un'entità può diventare proibitivo
 - Tutte le "vecchie" locazioni devono conservare informazione relativa all'entità
 - Basta un collegamento interrotto o corrotto per rendere l'entità irraggiungibile
 - Realizzato da una catena di coppie SSP < *proxy, skeleton* > in un sistema ad oggetti distribuiti
 - *Proxy (stub)* → riferimento allo *skeleton* finale od intermedio
 - *Skeleton (scion)* → riferimento locale all'oggetto oppure al suo *proxy* locale

Corso di Laurea Specialistica in Informatica, Università di Padova 18/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 7

Un oggetto che cambia locazione lascia un *proxy* al suo vecchio indirizzo ed installa uno *skeleton* che punta a se nella nuova locazione. Questa migrazione è del tutto trasparente al cliente.

La richiesta viaggia verso l'oggetto, non l'indirizzo verso il cliente!

Il sistema SSP (*stub-scion-pairs*) di forwarding

Corso di Laurea Specialistica in Informatica, Università di Padova 19/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 8

Il sistema SSP si presta a 2 ottimizzazioni

- La richiesta include l'identificatore del *proxy* iniziale
- La risposta include l'indirizzo attuale dell'oggetto
- Questo consente di creare un cortocircuito tra cliente ed oggetto remoto

Corso di Laurea Specialistica in Informatica, Università di Padova 20/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 9

Soluzione migliore: *home location*

- Tecnica utilizzata per supportare indirizzi IP mobili
- Ogni utente mobile ha un IP fisso al quale corrisponde un *home agent*
- Quando l'utente si sposta su un'altra rete riceve un nuovo indirizzo temporaneo (*care-of*) che viene registrato presso l'*home agent*
- L'*home agent* gira ogni pacchetto indirizzato all'utente verso il suo indirizzo *care-of* e ne informa il mittente
- Grande trasparenza, ma al costo di un contatto obbligatorio con la *home location* del destinatario

Corso di Laurea Specialistica in Informatica, Università di Padova 21/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 10

Soluzione ad approccio gerarchico 1/4

- La rete di interconnessione è suddivisa in una collezione di **domini** (similmente all'organizzazione del DNS)
 - Un singolo dominio copre l'intera rete
 - Ogni dominio può essere decomposto in sotto-domini
 - Un dominio non decomposto (terminale) è detto foglia e corrisponde ad un ben definito aggregato (p.es. una rete locale, una cella)
- Ogni dominio ha un **nodo repertorio** che conosce tutte le entità presenti in esso
 - Il nodo repertorio del dominio di livello più alto è detto nodo radice e conosce tutte le entità dell'intero sistema

Corso di Laurea Specialistica in Informatica, Università di Padova 22/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 11

Soluzione ad approccio gerarchico 2/4

- Ogni entità E presente in un dominio D è rappresentata da una voce (*location record*) nel nodo repertorio N di D
 - La voce di E nel dominio foglia di appartenenza contiene l'indirizzo di E in D
 - La voce di E nel dominio di livello immediatamente superiore a (i.e. contenente) D avrà solo un riferimento puntatore ad N
- Il nodo radice conterrà una voce (*name record*) per ogni entità del sistema, la quale conterrà l'inizio di una catena di puntatori a nodi repertori fino a quello una cui voce (*location record*) contiene l'indirizzo dell'entità

Corso di Laurea Specialistica in Informatica, Università di Padova 23/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 12

Soluzione ad approccio gerarchico 3/4

- Il *look-up* di E ha inizio nel nodo repertorio del dominio D0 di residenza del cliente
 - Se esso non contiene una voce per E, allora E non si trova in D0
 - In quel caso la richiesta viene inviata al nodo repertorio del dominio D1 "padre" di D0, che è più ampio (ossia conosce più entità) di D0, e così via fino a che si una voce per E venga trovata nel nodo repertorio del dominio M
 - Ciò significa che E si trova in M, dove verrà localizzata seguendo a ritroso la catena di riferimento, fino all'indirizzo di E nel suo dominio foglia
- In questo modo il *look-up* sfrutta la località dei riferimenti
 - La fase di "risalita" della ricerca avviene in domini concentricamente più ampi, nel caso peggiore fino al nodo radice, per poi da lì ridiscendere

Corso di Laurea Specialistica in Informatica, Università di Padova 24/34

Sistemi distribuiti: gestione dei nomi

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 13

Node has no record for E, so that request is forwarded to parent

Node knows about E, so request is forwarded to child

La modalità di look-up concentrica, ascendente e discendente

Look-up request

Domain D

Corso di Laurea Specialistica in Informatica, Università di Padova

25/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità mobili - 14

□ **Soluzione ad approccio gerarchico** 4/4

- Concentrare tutte le voci nel nodo radice pone problemi di scalabilità rendendolo collo di bottiglia del sistema
- Convienne partizionarne la conoscenza, ma il problema diventa deve localizzarne le parti
 - L'uso di calcolo parallelo o distribuito trasferisce il collo di bottiglia dall'elaborazione alla comunicazione
 - Meglio usare più (sotto-)nodi sparsi uniformemente in domini dell'intera rete
 - In questo caso il problema diventa la determinazione dei cammini minimi per localizzare il (sotto-)nodo responsabile dell'informazione richiesta

Corso di Laurea Specialistica in Informatica, Università di Padova

26/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità non (più) riferibili - 1

□ Un'entità che non possa (più) essere riferita è inutile al sistema e dovrebbe essere rimossa

- *Garbage collection*

□ Talvolta (di rado) la rimozione può essere esplicita → "l'ultimo" processo ad usare una data entità può rimuoverla

- Come sapere quale è "l'ultimo" processo in un sistema distribuito?

Corso di Laurea Specialistica in Informatica, Università di Padova

27/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità non (più) riferibili - 2

□ L'insieme dei riferimenti tra oggetti è un grafo diretto nel quale gli oggetti sono nodi e gli archi sono riferimenti

□ Uno specifico sottoinsieme di nodi non riferiti ma capaci di riferire è detto *root set* (p.es. servizi di sistema, utenti)

□ Gli oggetti non riferiti direttamente od indirettamente dal *root set* vanno rimossi

Corso di Laurea Specialistica in Informatica, Università di Padova

28/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità non (più) riferibili - 3

Root set

Entities forming an unreachable cycle

Reachable entity from the root set

Unreachable entity from the root set

Entità radice, raggiungibili e non raggiungibili

Corso di Laurea Specialistica in Informatica, Università di Padova

29/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità non (più) riferibili - 4

□ **Tecniche di rimozione: contatore dei riferimenti**

- Funziona bene in sistemi uniprocessore, ma è ostacolata dai possibili errori di comunicazione nei sistemi distribuiti
- L'oggetto distribuito mantiene il proprio contatore dei riferimenti nel suo *skeleton*
- Ogni *proxy* creato presso nodi utente invia una notifica di incremento allo *skeleton*, che ne conferma la ricezione
 - La mancata ricezione di conferma (*acknowledgement*) non implica il mancato incremento → rischio di duplicazione
 - Il passaggio di un riferimento remoto da un oggetto all'altro deve causare un incremento
- Ogni rimozione di *proxy* comporta l'invio di una notifica di decremento allo *skeleton*, che può però andare perduta
 - L'oggetto resta in vita anche se è privo di utenti

Corso di Laurea Specialistica in Informatica, Università di Padova

30/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità non (più) riferibili - 5

- Altre grave problema di questa tecnica è causato dalla *race condition* tra incrementi e decrementi
- Il problema sparisce usando solo decrementi
 - Alla creazione dell'oggetto remoto, al suo *skeleton* vengono assegnati un peso totale ($RC=2^n$) ed un peso parziale ($RW=2^m$)
 - Inizialmente $n=m$
 - Ad ogni creazione di un riferimento, $1/2$ RW dello *skeleton* viene ceduto al *proxy* corrispondente, con l'invariante $RC = \sum(RW)$
 - Ad ogni passaggio di riferimento, il *proxy* emittente cede $1/2$ del suo peso al destinatario
 - Ad ogni rimozione di un riferimento, il peso del corrispondente *proxy* viene sottratto ad RC dello *skeleton*
 - Quando RC dello *skeleton* va a 0 l'oggetto viene rimosso

Corso di Laurea Specialistica in Informatica, Università di Padova 31/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità non (più) riferibili - 6

Ala creazione dell'oggetto remoto:
 $RC = 2^n$; $RW(S) = RC$

Ala creazione del primo riferimento:
 $RW(P) = RW(S)/2$
 $RW(S) = RW(S)/2$
 $RC(S) = \sum RW$

Al passaggio di riferimento:
 $RW(P2) = RW(P1)/2$
 $RW(P1) = RW(P1)/2$
 $RC(S) = \sum RW$

Total and partial weight at skeleton remain the same

Corso di Laurea Specialistica in Informatica, Università di Padova 32/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità non (più) riferibili - 7

- Il problema è ora che ogni oggetto remoto consente solo N riferimenti (per N fissato)
 - Quando RW di *skeleton* e/o di *proxy* non è più dimezzabile (restituendo un intero) non sono più consentiti riferimenti
- Risolto tramite variante del *forwarding*
 - Una cella di indirizzazione (IC) rileva $RW=1$, ma riceve $RC=2^n$, che ripartisce tra i nuovi riferimenti come se fosse un nuovo oggetto (esempio $n = 3$)

Corso di Laurea Specialistica in Informatica, Università di Padova 33/34

Università degli Studi di Padova

Sistemi distribuiti: gestione dei nomi

Entità non (più) riferibili - 8

- Lista dei riferimenti (usata da Java RMI)
 - Un processo che crea un riferimento remoto deve prima identificarsi presso lo *skeleton* e, ricevuta conferma (*ack*) crea il *proxy*
 - Lo *skeleton* mantiene la lista dei riferimenti
 - Un processo che riceve un riferimento remoto (e dunque crea un *proxy* locale) deve fare lo stesso
 - Tecnica più informativa del conto dei riferimenti, che è anonima
 - Ugualmente esposta alla *race condition* tra inserzioni e rimozioni di riferimenti

Corso di Laurea Specialistica in Informatica, Università di Padova 34/34