


Gestione dei nomi

Anno accademico 2008/9
Sistemi Concorrenti e Distribuiti

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

SCD

Corso di Laurea Magistrale in Informatica, Università di Padova 1/40




Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 1

- Le **entità** di un sistema distribuito devono avere denotazioni che le rendano fruibili
 - Per riferimento, invocazione di servizio, ecc.
- Un **nome** è una stringa che riferisce entità
- I **punti d'accesso** (*access point*) sono entità speciali il cui nome è un **indirizzo**
- Una entità può avere più punti di accesso
 - Più "punti di vista" simultanei o anche diversi nel tempo

Corso di Laurea Magistrale in Informatica, Università di Padova 2/40



Sistemi distribuiti: gestione dei nomi

Esempio

- **Persona = entità di sistema**
 - Un servizio
- **Il suo telefono = access point del servizio**
 - Il dispositivo (*server*) che fornisce quel servizio
- **Il numero di telefono = nome (indirizzo) dell'access point di quel servizio**
 - L'indirizzo del *server* è il suo *end point* di livello trasporto

Corso di Laurea Magistrale in Informatica, Università di Padova 3/40



Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 2

- **Trattare gli indirizzi come nomi speciali**
 - 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
 - Una stessa entità può avere più punti di accesso
- **Vogliamo nomi di entità indipendenti dai loro indirizzi (*location independence*)**

Corso di Laurea Magistrale in Informatica, Università di Padova 4/40



Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 3

- **Certi nomi designano entità in modo univoco**
 - Per questi nomi usiamo **identificatori**
 - Non tutti i nomi sono identificatori
 - Alcuni nomi devono essere "trattabili" (*human-friendly*) ma non indirizzi e identificatori
- **Un vero identificatore**
 - Denota al più una singola entità
 - Non è riusabile
 - Una stessa entità non può averne più di uno
 - Così che, con essi, identificatori diversi designano entità diverse
 - Un numero di telefono fisso non un è vero identificatore di una entità "persona" perché può essere riassegnato!
- **Nomi diversi per una stessa entità sono detti alias**

Corso di Laurea Magistrale in Informatica, Università di Padova 5/40



Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 4

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

Corso di Laurea Magistrale in Informatica, Università di Padova 6/40

Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 5

Lo spazio dei nomi

- 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 a un nodo repertorio (*directory*)
 - Nome globale se interpretato sempre rispetto alla stessa *directory*
 - Nome locale se l'interpretazione dipende dal contesto d'uso
- I grafi dei nomi sono spesso aciclici ma non tutti

Corso di Laurea Magistrale in Informatica, Università di Padova 7/40

Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 6

Risoluzione dei nomi

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

Corso di Laurea Magistrale in Informatica, Università di Padova 8/40

Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 7

Grafo dei nomi con nodo radice unico e di nome implicito

Corso di Laurea Magistrale in Informatica, Università di Padova 9/40

Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 8

La risoluzione dei nomi è possibile anche su più spazi dei nomi uniti trasparentemente

- Il principio del *mount* su un *file system*
 - 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 serve
 - Il nome del protocollo di accesso al nodo che lo ospita
 - Il nome del *servente* che lo gestisce
 - Il nome della radice designata in esso

Corso di Laurea Magistrale in Informatica, Università di Padova 10/40

Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 9

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

- Livello globale → i nodi di più alto livello
 - Radici e loro figli (informazione generalmente stabile)
- Livello amministrativo → i nodi repertorio che sono amministrati da una singola responsabilità
 - Ciascun nodo repertorio rappresenta gruppi di entità appartenenti alla stessa organizzazione o unità amministrativa
- Livello gestionale → i nodi che possono cambiare frequentemente
 - Ciascun nodo gestito insieme dall'utente e dall'amministratore di sistema

Corso di Laurea Magistrale in Informatica, Università di Padova 11/40

Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 10

Corso di Laurea Magistrale in Informatica, Università di Padova 12/40

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** → il *name server* interrogato (quale?) fornisce la migliore risposta in suo possesso senza fare interrogazioni
 - Il *resolver* determina il nome completo dell'entità e lo fornisce al *name server* radice
 - Il cliente svolge più lavoro del server → non conviene fare *caching* presso il cliente
 - **Ricorsiva** → il *name server* interrogato (quale?) interroga altri *name server* per costruire la risposta
 - Il *name server* interrogato riceve il nome completo e fornisce la risposta finale
 - Il server svolge più lavoro del cliente → conviene fare *caching* presso il cliente

Corso di Laurea Magistrale in Informatica, Università di Padova 13/40

Sistemi distribuiti: gestione dei nomi

Risoluzione iterativa

Corso di Laurea Magistrale in Informatica, Università di Padova 14/40

Sistemi distribuiti: gestione dei nomi

Risoluzione ricorsiva

In generale NON sarà il NS radice!

Corso di Laurea Magistrale in Informatica, Università di Padova 15/40

Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 12

- **Servizi di repertorio (*directory services*)**
 - Realizzano sistemi di denominazione basati su attributi piuttosto che su nomi strutturati (come fa invece il DNS)
 - *Directory service* tratta spazi di nomi come insiemi di <attributo, valore>
 - *Naming system* tratta spazi di nomi strutturati
 - Dalla combinazione delle due tecniche ha origine LDAP (*lightweight directory access protocol*)
 - Derivante da OSI X.500 con ampi miglioramenti prestazionali
 - Ogni istanza di servizio LDAP gestisce una DIB (*directory information base*) con campi con attributi dal nome unico
 - L'architettura complessiva LDAP risulta molto simile a quella del DNS però anche più sofisticata e potente

Corso di Laurea Magistrale in Informatica, Università di Padova 16/40

Sistemi distribuiti: gestione dei nomi

LDAP – 1

Directory entry di DIB (paragonabile al *resource record* del DNS)

Attribute	Abbr.	Value
Country	C	NL
Locality	L	Amsterdam
Organization	O	Vrije Universiteit
OrganizationalUnit	OU	Comp. Sc.
CommonName	CN	Main server
Mail_Server	—	137.37.20.3, 130.37.24.6, 137.37.20.10
FTP_Server	—	130.37.20.20
WWW_Server	—	130.37.20.20

Corso di Laurea Magistrale in Informatica, Università di Padova 17/40

Sistemi distribuiti: gestione dei nomi

LDAP – 2

Corso di Laurea Magistrale in Informatica, Università di Padova 18/40

Sistemi distribuiti: gestione dei nomi

Entità mobili – 1

- ❑ I nomi dei livelli globale e 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 Magistrale in Informatica, Università di Padova 19/40

Sistemi distribuiti: gestione dei nomi

Entità mobili – 2

- ❑ Cambiare i nomi usati come identificatori
 - Invalida i *symbolic link* che li utilizzano
- ❑ Modificare lo spazio dei nomi del nodo repertorio di origine
 - Associando un nuovo indirizzo al vecchio nome
 - *Look-up* veloce
 - Ogni successivo aggiornamento (su nodo di origine) ha costo elevato
 - Associando un nuovo nome al valore del vecchio nome
 - = *symbolic link*
 - *Look-up* più lento
 - Facile aggiornamento tramite nuovo *symbolic link* locale al costo di una ulteriore indirizzazione

Corso di Laurea Magistrale in Informatica, Università di Padova 20/40

Sistemi distribuiti: gestione dei nomi

Entità mobili – 3

- ❑ Entrambe le soluzioni sono inadeguate per sistemi distribuiti a larga scala
- ❑ Convien separare i nomi dagli indirizzi
 - Non come nel DNS di *Internet*
- ❑ Gli identificatori sono adatti a questo scopo
 - Da nome utente a identificatore → *naming service*
 - Da identificatore a indirizzo → *location service*

Corso di Laurea Magistrale in Informatica, Università di Padova 21/40

Sistemi distribuiti: gestione dei nomi

Entità mobili – 4

Corso di Laurea Magistrale in Informatica, Università di Padova 22/40

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: *broadcast*
 - ARP (*Address Resolution Protocol*) restituisce l'indirizzo di livello collegamento dati in rete locale corrispondente a un indirizzo IP in ingresso
 - Troppo oneroso per reti vaste
 - Meglio *multicast* verso gruppi specifici (anche dinamici) di nodi coinvolti

Corso di Laurea Magistrale in Informatica, Università di Padova 23/40

Sistemi distribuiti: gestione dei nomi

Entità mobili – 6

- ❑ Soluzione meno brutale: *forwarding*
 - Simile a 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 SSP, coppie <*proxy, skeleton*>, in un sistema a oggetti distribuiti
 - *Proxy (stub)* → riferimento allo *skeleton* finale o intermedio
 - *Skeleton (scafo)* → riferimento locale all'oggetto oppure al suo *proxy* locale
 - *Scafo* = discendente in linea ereditaria

Corso di Laurea Magistrale in Informatica, Università di Padova 24/40

Sistemi distribuiti: gestione dei nomi

Entità mobili – 7

Un oggetto che cambia locazione lascia un *proxy* al suo vecchio indirizzo e 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 Magistrale in Informatica, Università di Padova 25/40

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 e oggetto remoto

Corso di Laurea Magistrale in Informatica, Università di Padova 26/40

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 cui 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 al costo di un contatto obbligatorio con la *home location* del destinatario

Corso di Laurea Magistrale in Informatica, Università di Padova 27/40

Sistemi distribuiti: gestione dei nomi

Entità mobili – 10

Soluzione ad approccio gerarchico I

- Rete di interconnessione suddivisa in una collezione di domini (similmente al 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 a 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 Magistrale in Informatica, Università di Padova 28/40

Sistemi distribuiti: gestione dei nomi

Entità mobili – 11

Soluzione ad approccio gerarchico II

- Ogni entità E di dominio D è rappresentata da un *location record* nel nodo repertorio N di D
 - Il *location record* di E in N contiene l'indirizzo di E in D
 - Alla voce E nel dominio di livello immediatamente superiore a D (che contiene D) corrisponde solo un riferimento puntatore a N
- Il nodo radice conterrà un *name record* per ogni entità di sistema
 - Il *name record* conterrà l'inizio di una catena di puntatori a nodi repertori fino a quello un cui *location record* contiene l'indirizzo dell'entità

Corso di Laurea Magistrale in Informatica, Università di Padova 29/40

Sistemi distribuiti: gestione dei nomi

Entità mobili – 12

Soluzione ad approccio gerarchico III

- Il *look-up* di E ha inizio nel nodo repertorio del dominio D_0 di residenza del cliente che lo richiede
 - Se D_0 non contiene informazioni per E allora E non si trova in D_0
 - La richiesta viene allora inviata al nodo repertorio del dominio D_1 "genitore" di D_0 (che per definizione conosce più entità di D_0) e così via fino a che un *record* per E venga trovato nel nodo repertorio del dominio D_N
 - Allora E si trova in D_N dove verrà localizzato 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 Magistrale in Informatica, Università di Padova 30/40

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

Modalità di *look-up* concentrica, ascendente e discendente

Look-up request

Domain D

Corso di Laurea Magistrale in Informatica, Università di Padova 31/40

Sistemi distribuiti: gestione dei nomi

Entità mobili – 14

□ Soluzione ad approccio gerarchico IV

- Concentrare tutti i *name record* nel nodo radice pone gravi problemi di scalabilità
 - Rendendolo collo di bottiglia del sistema
- Meglio **partizionare la conoscenza**
- Il problema diventa decidere dove 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 Magistrale in Informatica, Università di Padova 32/40

Sistemi distribuiti: gestione dei nomi

Entità non riferibili – 1

□ Un'entità che non possa essere riferita è inutile al sistema e va rimossa

- *Garbage collection*

□ La rimozione può essere esplicita

- "L'ultimo" processo a usare una data entità può rimuoverla
- Ma come individuare "l'ultimo" processo in un sistema distribuito?

Corso di Laurea Magistrale in Informatica, Università di Padova 33/40

Sistemi distribuiti: gestione dei nomi

Entità non 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*

- Esempio: servizi di sistema, utenti

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

Corso di Laurea Magistrale in Informatica, Università di Padova 34/40

Sistemi distribuiti: gestione dei nomi

Entità non 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 Magistrale in Informatica, Università di Padova 35/40

Sistemi distribuiti: gestione dei nomi

Entità non riferibili – 4

□ Tecniche di rimozione: contatore dei riferimenti

- Adatta ai sistemi centralizzati ma ostacolata dai possibili errori di comunicazione nei sistemi distribuiti
- L'oggetto remoto mantiene un contatore dei riferimenti
 - Nel suo *skeleton*
- Ogni *proxy* creato presso nodi utente invia una notifica di incremento allo *skeleton* con conferma di 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 invio di notifica di decremento allo *skeleton* che può però andare perduta
 - L'oggetto potrebbe restare in vita anche se privo di utenti

Corso di Laurea Magistrale in Informatica, Università di Padova 36/40

Sistemi distribuiti: gestione dei nomi

Entità non riferibili – 5

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

Corso di Laurea Magistrale in Informatica, Università di Padova

37/40

Sistemi distribuiti: gestione dei nomi

Entità non riferibili – 6

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

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

Al passaggio di riferimento:
 $WR(P_{i+1}) = WR(P_i)/2$
 $WR(P_i) = WR(P_{i+1})/2$
 $RC(S) = \sum WR$

Corso di Laurea Magistrale in Informatica, Università di Padova

38/40

Sistemi distribuiti: gestione dei nomi

Entità non riferibili – 7

- ❑ In questo modo ogni oggetto remoto consente solo N riferimenti per N fissato
 - Per WR (di *skeleton* e/o di *proxy*) non più dimezzabile restituendo un intero non sono più consentiti riferimenti
- ❑ Risolto tramite variante del *forwarding*
 - Una cella di indirection (IC) rileva $WR=1$ ma riceve $RC=2^n$ che ripartisce tra i nuovi riferimenti come se fosse un nuovo oggetto (esempio $n = 3$)

Corso di Laurea Magistrale in Informatica, Università di Padova

39/40

Sistemi distribuiti: gestione dei nomi

Entità non riferibili – 8

- ❑ Lista dei riferimenti
 - Usata da Java RMI per GC distribuito
 - Un processo che crea o riceve un riferimento remoto deve prima identificarsi presso lo *skeleton*
 - Ricevuta conferma (*ack*) crea il *proxy*
 - Lo *skeleton* dell'oggetto remoto mantiene la lista dei riferimenti
 - Riferimento remoto considerato in "affitto" (*leasing*)
 - Rimosso automaticamente in assenza d'uso
 - Prima che il GC locale rimuova un *proxy* ne comunica la cancellazione allo *skeleton* corrispondente
 - Gestione dei riferimenti nello *skeleton* realizzate da RPC con semantica *at-most-once*
- ❑ Più informativa del *reference count* che è anonimo
 - Rischio di *race condition* tra inserzioni e rimozioni di riferimenti

Corso di Laurea Magistrale in Informatica, Università di Padova

40/40

UniPD - SCD 2008/9 - Sistemi Concorrenti e Distribuiti

7