



Gestione dei nomi

Anno accademico 2010/11
Sistemi Concorrenti e Distribuiti

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



Corso di Laurea Magistrale in Informatica, Università di Padova 1/38



Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 1

- ❑ Le **entità** di un sistema distribuito devono avere denotazioni che le rendano note
 - 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**
 - Un nome di entità che non varia con l'indirizzo è detto *location-independent*
- ❑ Un'entità può offrire più punti di accesso
 - Più "punti di vista" simultanei o anche diversi nel tempo

Corso di Laurea Magistrale in Informatica, Università di Padova 2/38



Sistemi distribuiti: gestione dei nomi

Esempio

- ❑ **Persona = entità di sistema**
 - Il suo ruolo pubblico è offrire un servizio
- ❑ **Telefono = access point del servizio**
 - Ciò che va contattato per ottenere quel servizio
 - Quindi ne è il *server*
- ❑ **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 3/38



Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 2

- ❑ **Gli indirizzi sono nomi speciali**
 - Conviene decomporre la corrispondenza <entità-indirizzo> in 2 relazioni 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 4/38



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 *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
 - Identificatori diversi designano entità diverse
 - Un numero di telefono fisso non è un 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 5/38



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 directory** → da cui dipartono archi etichettati con un nome, contiene una tabella di corrispondenze <etichetta d'arco, identificatore di nodo>
 - **Nodo 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 **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 6/38

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 *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

7/38

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 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 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

8/38

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

9/38

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 dei **servente** che lo gestisce
 - Il nome della **radice** designata in esso

Corso di Laurea Magistrale in Informatica, Università di Padova

10/38

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 *directory* amministrati da una singola autorità
 - Ciascun nodo *directory* rappresenta gruppi di entità appartenenti alla stessa organizzazione o unità amministrativa
- **Livello gestionale** → i nodi che possono cambiare frequentemente
 - Ciascun nodo gestito dall'utente e dall'amministratore di sistema

Corso di Laurea Magistrale in Informatica, Università di Padova

11/38

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 Magistrale in Informatica, Università di Padova

12/38

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 servente → 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 servente svolge più lavoro del cliente → conviene fare *caching* presso il cliente

Corso di Laurea Magistrale in Informatica, Università di Padova 13/38

Sistemi distribuiti: gestione dei nomi

Risoluzione iterativa

Corso di Laurea Magistrale in Informatica, Università di Padova 14/38

Sistemi distribuiti: gestione dei nomi

Risoluzione ricorsiva

In generale NON sarà il NS radice!

Corso di Laurea Magistrale in Informatica, Università di Padova 15/38

Sistemi distribuiti: gestione dei nomi

Denominazione di entità – 12

- **Directory service**
 - Realizzano sistemi di denominazione basati su attributi piuttosto che su nomi strutturati
 - Come fa invece il DNS per il quale si parla di *naming service*
 - 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 LDAP è molto simile a quella DNS ma più sofisticata e potente

Corso di Laurea Magistrale in Informatica, Università di Padova 16/38

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 17/38

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 directory di origine**
 - Soluzione 1: associare nuovo indirizzo a vecchio nome
 - *Look-up* veloce
 - Ogni successivo aggiornamento (su nodo di origine) ha costo elevato
 - Soluzione 2: associare nuovo nome a valore del vecchio nome
 - = *symbolic link*
 - *Look-up* più lento
 - Facile aggiornamento tramite nuovo *symbolic link* locale al costo di ulteriore indirizzatura

Corso di Laurea Magistrale in Informatica, Università di Padova 18/38

Sistemi distribuiti: gestione dei nomi

Entità mobili – 3

- Entrambe le soluzioni sono inadeguate per sistemi distribuiti a larga scala
- Conviene 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 19/38

Sistemi distribuiti: gestione dei nomi

Entità mobili – 4

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

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

Corso di Laurea Magistrale in Informatica, Università di Padova 20/38

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 21/38

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 (scion)* → riferimento locale all'oggetto oppure al suo *proxy* locale
 - *Scion* = discendente in linea ereditaria

Corso di Laurea Magistrale in Informatica, Università di Padova 22/38

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 23/38

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
 - Ciò crea un cortocircuito tra cliente e oggetto remoto

Corso di Laurea Magistrale in Informatica, Università di Padova 24/38

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 25/38

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 26/38

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 *directory* 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 in cui *location record* contiene l'indirizzo dell'entità

Corso di Laurea Magistrale in Informatica, Università di Padova 27/38

Sistemi distribuiti: gestione dei nomi

Entità mobili – 12

□ **Soluzione ad approccio gerarchico III**

- Il *look-up* di E ha inizio nel nodo *directory* 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 *directory* 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 *directory* 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 28/38

Sistemi distribuiti: gestione dei nomi

Entità mobili – 13

Look-up request

Domain D

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* concentrata, ascendente e discendente

Corso di Laurea Magistrale in Informatica, Università di Padova 29/38

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
 - Centralizzazione \Rightarrow collo di bottiglia
- Meglio partizionare la conoscenza
- 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 30/38

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 31/38

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 32/38

Sistemi distribuiti: gestione dei nomi

Entità non riferibili – 3

Entità radice, raggiungibili e non raggiungibili

Corso di Laurea Magistrale in Informatica, Università di Padova 33/38

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 nel suo *skeleton* un contatore dei riferimenti
 - Ogni *proxy* creato presso nodi cliente invia una notifica di incremento allo *skeleton* (con conferma di ricezione)
 - La mancata ricezione di conferma (*acknowledgment*) 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 34/38

Sistemi distribuiti: gestione dei nomi

Entità non riferibili – 5

- Altro grave problema del *reference counting* è la *race condition* tra incrementi e decrementi
- Meglio usare 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 35/38

Sistemi distribuiti: gestione dei nomi

Entità non riferibili – 6

(a) Alla creazione dell'oggetto remoto: $RC = 2^n$; $WR(S) = RC$
 (b) Alla creazione del primo riferimento: $WR(P) = WR(S)/2$; $WR(S) = WR(S)/2$; $RC(S) = \sum WR$
 (c) Al passaggio di riferimento: $WR(P_{P2}) = WR(P_{P1})/2$; $WR(P_{P1}) = WR(P_{P1})/2$; $RC(S) = \sum WR$

Corso di Laurea Magistrale in Informatica, Università di Padova 36/38

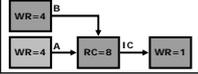


Sistemi distribuiti: gestione dei nomi

Entità non riferibili – 7

- ❑ **Ma 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 indirectione (IC) rileva WR=1 ma riceve RC=2ⁿ che ripartisce tra i nuovi riferimenti come se fosse un nuovo oggetto (esempio n = 3)





Corso di Laurea Magistrale in Informatica, Università di Padova
37/38



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
 - Il riferimento remoto viene considerato in "affitto" (*leasing*) e rimosso 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
38/38