Aspetti software delle reti

Le prime reti furono progettate cominciando dall'hardware e sviluppando il software solo successivamente, quasi come se esso fosse un'accessoria appendice dell'hardware.Questo approccio non funziona più. Il SW di rete è oggi altamente strutturato. Esaminiamo ora, a grandi linee, tale strutturazione.

Gerarchie di protocollo

Per ridurre la complessità di progetto, le reti sono in generale organizzate a livelli, ciascuno costruito sopra il precedente. Fra un tipo di rete ed un altra, possono essere diversi:

- il numero di livelli;
- i nomi dei livelli;
- il contenuto dei livelli;
- le funzioni dei livelli.

Comunque un principio generale è sempre rispettato:

- lo scopo di un livello è offrire certi servizi ai livelli più alti, nascondendo i dettagli sul come tali servizi siano implementati.

Il livello n su un host porta avanti una conversazione col livello n su di un'altro host. Le regole e le convenzioni che governano la conversazione sono collettivamente indicate col termine di protocollo di livello n.

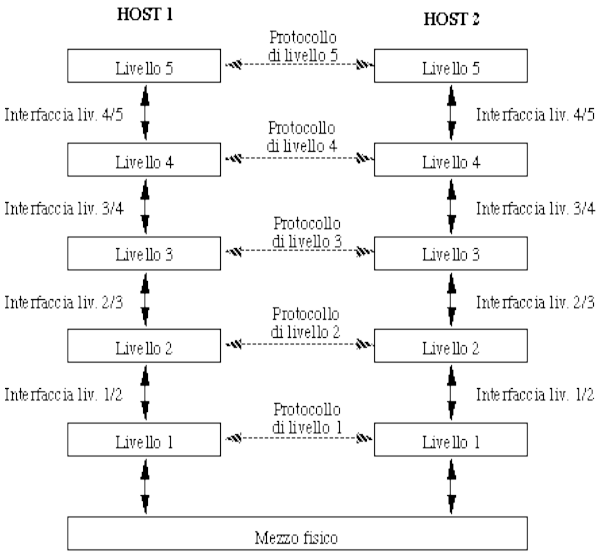

***Figura 1**: Dialogo fra peer entity*

Le entità (processi) che effettuano tale conversazione si chiamano peer entitiy (entità di pari livello).
Il dialogo fra due peer entity di livello n viene materialmente realizzato tramite i servizi offerti dal livello (n-1).
In realtà non c'è un trasferimento diretto dal livello n di host 1 al livello n di host 2. Ogni livello di host 1 passa i dati, assieme a delle informazioni di controllo, al livello sottostante.
Al di sotto del livello 1 c'è il mezzo fisico, attraverso il quale i dati vengono trasferiti da host 1 ad host 2.
Quando arrivano a host 2, i dati vengono passati da ogni livello (a partire dal livello 1) a quello superiore, fino a raggiungere il livello n.

Fra ogni coppia di livelli adiacenti è definita una interfaccia, che caratterizza:

- le operazioni primitive che possono essere richieste al livello sottostante;
- i servizi che possono essere offerti dal livello sottostante.

I vantaggi di una buona progettazione delle interfacce sono:

- minimizzazione delle informazioni da trasferire;
- possibilità di modificare l'implementazione del livello (ad es., ove le linee telefoniche venissero sostituite da canali satellitari) con una più attuale che offra gli stessi servizi.

Architettura di rete

L'insieme dei livelli e dei relativi protocolli è detto architettura di rete.
La specifica dell'architettura deve essere abbastanza dettagliata da consentire la realizzazione di SW e/o HW che, per ogni livello, rispetti il relativo protocollo.
Viceversa, i dettagli implementativi di ogni livello e le interfacce fra livelli non sono parte dell'architettura, in quanto sono nascosti all'interno di un singolo host.
E'quindi possibile che sui vari host della rete ci siano implementazioni che differiscono fra di loro anche in termini di interfacce fra livelli, purché ogni host implementi correttamente i protocolli previsti dall'architettura. In questo caso possono dialogare fra loro anche host aventi caratteristiche (processore, sistema operativo, costruttore) diverse.
Dunque, nell'ambito di una specifica architettura di rete, si ha che:

- tutti gli host devono contenere implementazioni conformi in termini di livelli e di protocolli;
- gli host possono contenere implementazioni che differiscono in termini di dettagli implementativi e di interfacce fra livelli;

Un'architettura di rete può essere:

- proprietaria;
- standard de facto;
- standard de iure.

Un'architettura proprietaria è basata su scelte indipendenti ed arbitrarie del costruttore, ed è generalmente incompatibile con architetture diverse. Nel senso più stretto del termine è un'architettura per la quale il costruttore non rende pubbliche le specifiche, per cui nessun altro può produrre apparati compatibili.
Esempi:

- IBM SNA (System Network Architecture)
- Digital Decnet Phase IV;
- Novell IPX;
- Appletalk.

Un'architettura standard de facto è un'architettura basata su specifiche di pubblico dominio (per cui diversi costruttori possono proporre la propria implementazione) che ha conosciuto una larghissima diffusione.
Esempi:

- Internet Protocol Suite (detta anche architettura TCP/IP).

Un'architettura standard de iure è un'architettura basata su specifiche (ovviamente di pubblico dominio) approvate da enti internazionali che si occupano di standardizzazione. Anche in questo caso ogni costruttore può proporne una propria implementazione.
Esempi:

- standard IEEE 802 per le reti locali;
- architettura OSI (Open Systems Interconnection);
- Decnet Phase V (conforme allo standard OSI).

L'insieme dei protocolli utilizzati su un host e relativi ad una specifica architettura di rete va sotto il nome di pila di protocolli (protocol stack). Si noti che un host può avere contemporaneamente attive più pile di protocolli.

Funzionamento del software di rete

Nel caso delle reti, la comunicazione fra le due entità di livello superiore avviene con una modalità che, almeno in linea di principio, è uguale in tutte le architetture di rete:

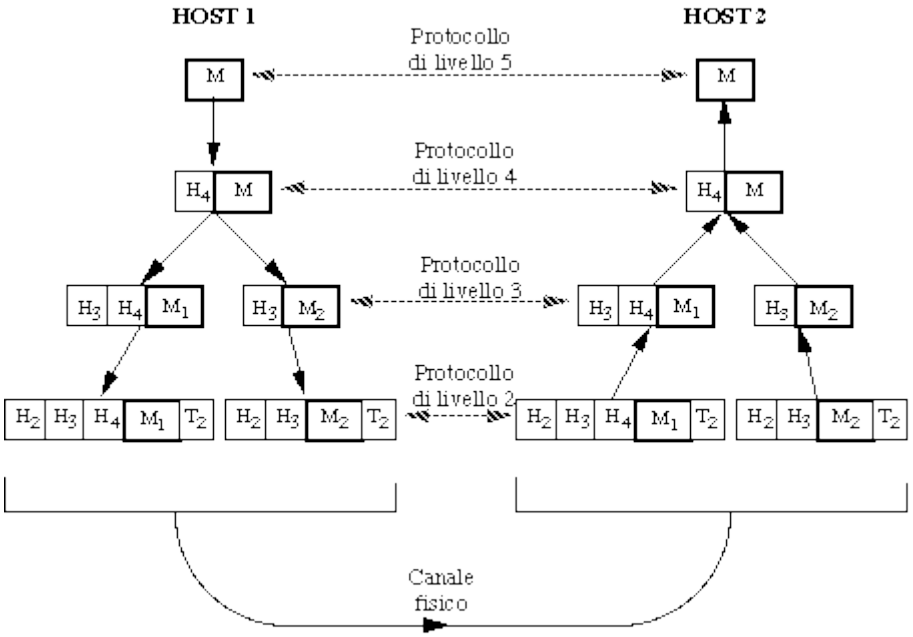

Figura 2: *Flusso dell'informazione fra peer entity.*

Vediamo cosa accade:

1. il programma applicativo (livello 5) deve mandare un messaggio M alla sua peer entity;
2. il livello 5 consegna M al livello 4 per la trasmissione;
3. il livello 4 aggiunge un suo header in testa al messaggio (talvolta si dice che il messaggio è inserito nella busta di livello 4); questo header contiene informazioni di controllo, tra le quali:
   - numero di sequenza del messaggio;
   - dimensione del messaggio;
   - time stamp;
   - priorità;
4. il livello 4 consegna il risultato al livello 3;
5. il livello 3 può trovarsi nella necessità di frammentare i dati da trasmettere in unità più piccole (pacchetti), a ciascuna delle quali aggiunge il suo header;
6. il livello 3 passa i pacchetti al livello 2;
7. il livello 2 aggiunge ad ogni pacchetto il proprio header (e magari un trailer) e lo spedisce sul canale fisico;
8. nella macchina di destinazione i pacchetti fanno il percorso inverso, con ogni livello che elimina (elaborandoli) l'header ed il trailer di propria competenza, e passa il resto al livello superiore.

Aspetti importanti sono i seguenti:

- le peer entity pensano concettualmente ad una comunicazione orizzontale fra loro, basata sul protocollo del proprio livello, mentre in realtà comunicano ciascuna col livello sottostante attraverso l'interfaccia fra i due livelli;
- spesso i livelli bassi sono implementati in hardware o firmware (per ragioni di efficienza). Nonostante questo, spesso gli algoritmi di gestione sono complessi.

Interfacce e servizi

La funzione di ogni livello è di offrire servizi al livello superiore.
Il livello inferiore è il service provider, quello superiore è il service user.
Un livello n che usufruisce dei servizi di livello (n-1) può, per mezzo di questi, a sua volta offrire al livello (n+1) i propri servizi.
I servizi sono disponibili ai SAP (Service Access Point). I SAP del livello n, o n-SAP, sono i punti di accesso nei quali il livello (n+1) può accedere ai servizi del livello n. Ogni n-SAP ha un indirizzo che lo identifica univocamente.

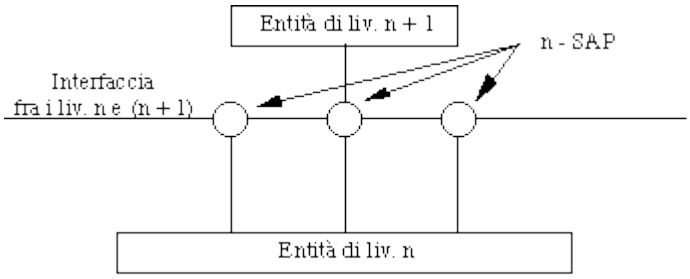

Figura 3: *Livelli adiacenti e service access point.*

Analogia col telefono:

- SAP: presa a muro del telefono;
- SAP address: numero telefonico che identifica quella presa.

L'informazione passata dal livello n al livello (n-1), attraverso il (n-1)-SAP, si dice PDU (Protocol Data Unit) di livello n, o n-PDU.
Essa, entrando nel livello (n-1), diventa una SDU (Service Data Unit) di livello (n-1), o (n-1)-SDU.
Entro il livello (n-1) viene aggiunta alla (n-1)-SDU una PCI (Protocol Control Information) di livello (n-1).
Il tutto diventa una (n-1)-PDU, che verrà passata al livello (n-2) attraverso un (n-2)-SAP.

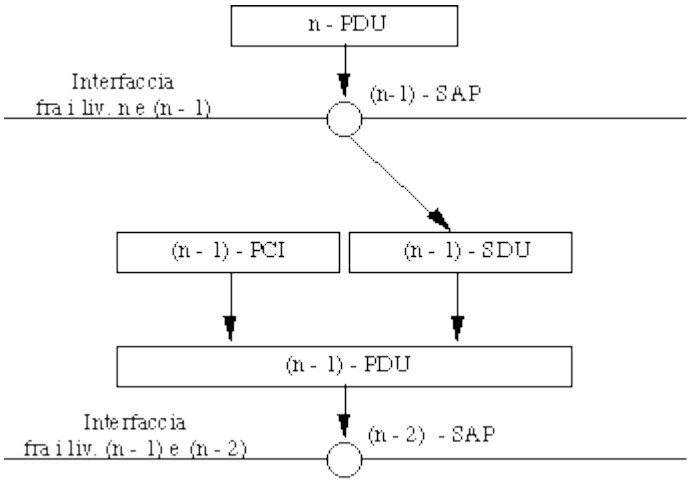

Figura 4: *Passaggio dell'informazione fra livelli.*

Nomi spesso usati per i PDU:

- segmento (o TPDU, Transport PDU) a livello transport;
- pacchetto (packet) a livello network;
- trama (frame) a livello data link.

Nome per il PCI:

- busta.

Servizi connection-oriented e connectionless

Ci sono due principali classi di servizi offerti da un livello a quello superiore:

- servizi connection-oriented;
- servizi connectionless.

Servizi connection-oriented

I servizi connection-oriented sono modellati secondo il sistema telefonico, dove per parlare con qualcuno si alza il telefono, si chiama, si parla e poi si riattacca. Ovvero:

1. si stabilisce una connessione;
2. si scambiano informazioni;
3. si rilascia la connessione.

Analogamente, un servizio connection-oriented si sviluppa in 3 fasi:

1. si stabilisce una connessione, cioé si crea con opportuni mezzi un "canale di comunicazione" fra la sorgente e la destinazione. La relativa attività tipicamente coinvolge un certo numero di elaboratori nel cammino fra sorgente e destinazione;
2. la connessione, una volta stabilita, agisce come un tubo digitale lungo il quale scorrono tutti i dati trasmessi, che arrivano nello steso ordine in cui sono partiti;
3. si rilascia la connessione (attività che coinvolge di nuovo tutti gli elaboratori sul cammino).

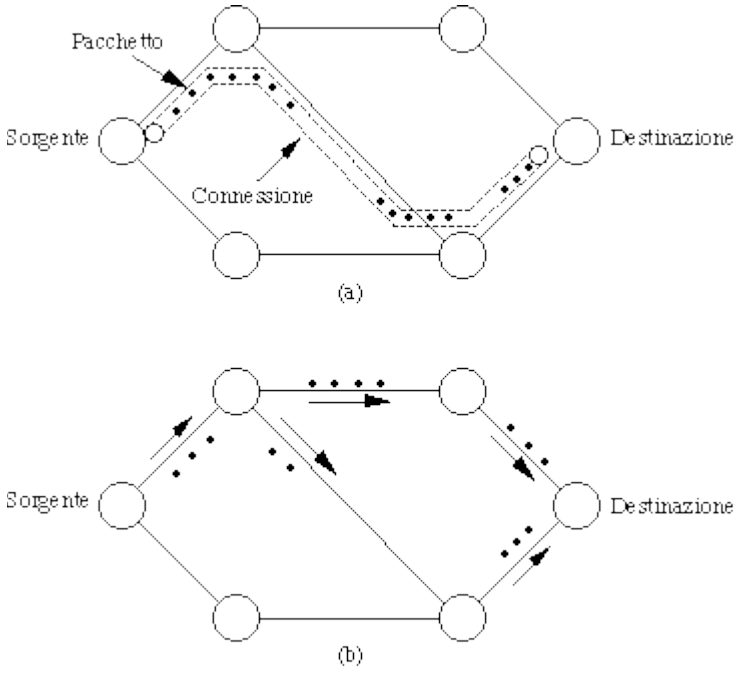

Figura 5: *Servizi connection-oriented (a) e connectionless (b).*

Servizi connectionless

I servizi connectionless sono modellati secondo il sistema postale: ogni lettera viaggia indipendentemente dalle altre; arriva quando arriva, e forse non arriva. Inoltre, due lettere con uguale mittente e destinatario possono viaggiare per strade diverse.

Analogamente, in un servizio connectionless, i pacchetti (PDU) viaggiano indipendentemente gli uni dagli altri, possono prendere strade diverse ed arrivare in ordine diverso da quello di partenza o non arrivare affatto.

La fase è una sola:

1. invio del pacchetto (corrisponde all'immissione della lettera nella buca).

Affidabilità del servizio

Un servizio è generalmente caratterizzato dall'essere o no affidabile (reliable).
Un servizio affidabile non perde mai dati, cioè assicura che tutti i dati spediti verranno consegnati al destinatario. Ciò generalmente richiede che il ricevente invii un acknowledgement (conferma) alla sorgente per ogni pacchetto ricevuto. Si introduce ovviamente overhead, che in certe situazioni può non essere desiderabile.
Viceversa, un servizio non affidabile non offre la certezza che i dati spediti arrivino effettivamente a destinazione.
Si noti che se un certo livello non offre nessun servizio affidabile, qualora tale funzionalità sia desiderata dovrà essere fornita da almeno uno dei livelli superiori (vedremo che ciò accade spesso).
Esempi:

- reliable connection oriented: trasferimento di file (non devono mancare pezzi e il file non deve essere "rimescolato");
- non reliable connection oriented: nelle trasmissioni isocrone (quali voce e video) le relazioni temporali fra i bit del flusso devono essere mantenute. È meglio qualche disturbo ogni tanto, piuttosto che interruzioni momentanee, ma avvertibili, del flusso di dati;
- non reliable connectionless (detto anche datagram service, da telegram): distribuzione di posta elettronica pubblicitaria, non importa se qualche messaggio si perde.
- reliable connectionless (detto anche acknowledged datagram service): si invia un breve messaggio e si vuole essere assolutamente sicuri che è arrivato.

Primitive di definizione del servizio

Un servizio di livello n è formalmente specificato da un insieme di primitive (cioé operazioni) che un'entità di livello (n+1) può adoperare per accedere al servizio. Esse possono indicare al servizio:

- l'azione da compiere (l'informazione viaggia da livello n al livello (n-1));
- cosa riportare in merito ad una azione effettuata dalla peer entity di livello n (l'informazione viaggia dal livello (n-1) al livello n).

Le primitive hanno vari parametri (mittente, destinatario, tipo del servizio richiesto, ecc.), che possono essere usati dalle peer entity per negoziare le caratteristiche della connessione. I dettagli della negoziazione fanno parte del protocollo.

Servizi vs. protocolli

Servizi e protocolli sono spesso confusi, ma sono concetti ben distinti.
Servizio: insieme di operazioni primitive che un livello offre al livello superiore. Come tali operazioni siano implementate non riguarda il livello superiore.
Protocollo: insieme di regole che governano il formato ed il significato delle informazioni (messaggi, frame, pacchetti) che le peer entity si scambiano fra loro. Le entità usano i protocolli per implementare i propri servizi.

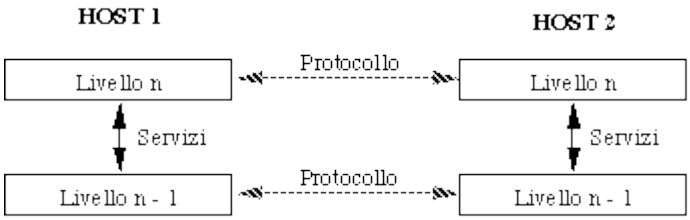

Figura 6: *Relazione fra protocolli e servizi.*

Aspetti di progetto dei livelli

Decisioni di progetto vanno prese, nei vari livelli, in merito a diverse problematiche. Le principali sono:

1. Meccanismi di identificazione di mittente e destinatario (cioé indirizzamento), in ogni livello.
2. Regole per il trasferimento dati (livelli bassi):
   - in una sola direzione (simplex connection);
   - in due direzioni ma non contemporaneamente (half-duplex connection).
   - in due direzioni contemporaneamente (full-duplex connection);
3. Meccanismi per il controllo degli errori di trasmissione; è possibile:
   - rilevarli oppure no;
   - correggerli oppure no;
   - avvertire il mittente oppure no.
4. Meccanismi per il mantenimento (o la ricostruzione) dell'ordine originario dei dati.
5. Meccanismi per regolare le velocità di sorgente e destinazione.
6. Decisioni sulla dimensione (minima o massima) dei messaggi da inviare, e su come eventualmente frammentarli.
7. Meccanismi di multiplexing di varie "conversazioni" su di un'unica connessione (se stabilire la connessione è costoso).
8. Meccanismi di routing dei messaggi se esistono più strade alternative, ed eventualmente di suddivisione di una "conversazione" su più connessioni contemporaneamente (per aumentare la velocità di trasferimento dei dati).