Network service models

Modelli di servizio

I servizi sono classificati in:

- Orientati alla conessione
- connection-less

Orientati alla connessione

Sono modelli come il **servizio telefonico**:

- Prima dello scambio di dati, **viene stabilita una connessione**
- I dati vengono scambiati
- La connessione viene **rilasciata**

Connection-less

Sono modellati come il **servizio postale**:

- I dati attraversano la rete attraverso **pacchetti indipendenti**
- L'ordine di trasmissione **non è necessariamente preservato** alla destinazione.

I servizi possono essere classificati in:

- **Affidabili** : garantiscono la consegna dei dati
- **Inaffidabili** : non è detto che i dati arrivino alla destinazione

Combinando diversi tipi di servizi possiamo avere:

- **Affidabili orientati alla connessione** : ad esempio il **trasferimento di files** ; il contenuto dei file **non deve essere alterato** , inoltre **i dati devono arrivare preservando l'ordine di trasmissione** .
- **inaffidabili Orientati alla connessione:** ad esempio la **comunicazione isocrona** (che avviene allo stesso momento), come la trasmissione di **voce e video** . La perdita dei dati è **tollerata** , affinchè sia garantita la **continuità** .
- **Affidabili senza connessione:** la comunicazione è basata su **piccoli messaggi** che **devono** arrivare a destinazione.
- **Inaffidabili senza connessione:** la comunicazione è basata su **piccoli messaggi** che **possono non** arrivare a destinazione.

Organizzazione di software per i sistemi di reti

Perchè è richiesto il software?

Accedere direttamente all'hardware **non è un approccio conveniente**. Questo perchè vorremmo usare un tipo di astrazione **ad alto livello**. Accedere a dispositivi network in maniera diretta è come accedere al disco attraverso le chiamate per leggere un settore specifico.

Per questo motivo, il **networking software** ci fornisce **un'interfaccia ad alto livello** per sviluppare applicazioni.

Come dovrebbe essere organizzato il software?

Una corretta progettazione del software di networking richiede:

- Di analizzare i **problemi di comunicazione** .
- Di decomporre i problemi in **sotto-problemi** .
- Di progettare ed **implementare software** per uno specifico **sotto-problema** .

Architettura di un software di networking

Il **modello a strati** è una soluzione al problema della complessità del **software di networking**. Il modello suggerisce di dividere il software **in strati**, ognuno devoto a **risolvere una parte del problema di comunicazione**. Gli strati presentano **diversi vincoli**, che facilitano la progettazione del software.

I software di networking possono presentare diversi

- Numeri di strati
- Nomi di strati
- Contenuti per strato
- Funzioni per strato

Architettura basata su strati

L'obbiettivo di uno strato è quello di offrire un **servizio specifico** agli strati superiori, nascondendo i **dettagli di implementazione** degli strati inferiori.

Lo strato **n** di un host comunica **solo con lo strato n** di un altro host (evidenziato in rosso nella foto).

Le entità che sono "capaci di parlare" sono chiamate **peer-entity**.

Le regole che controllano la conversazione sono chiamate **protocol of layer n**.

Il concetto di protocollo

Un **protocollo** definisce il formato e l'ordine dei messaggi scambiati tra due o più **entità comunicanti** e le azioni corrisponenti alla trasmissione e ricezione dei messaggi.

- **sintassi** : formato del messaggio
- **Semantica** : significato dei campi di messaggio

Architettura Network

Gli strati ed i protocolli annessi vengono chiamati **architettura della rete (network architecture)**.

Le specifiche dell'architettura dovrebbero essere sufficientemente dettagliate per permettere la corretta implementazione di SW e/o HW per ogni layer:
I protocolli dovrebbero essere specificati correttamente.

I dettagli di implementazione di ogni strato e le interfacce tra strati **non sono parte** dell'architettura di rete.

Diversi hosts possono avere diverse implementazioni, ma essi **devono implementare gli stessi protocolli**.

Layer interfaces

Non è presente una **comunicazione diretta** tra lo strato **n** di un host e lo strato **n** di un host differente:

- La comunicazione diretta è implementatta **attraversando tutti gli strati sottostanti.**
- Ogni volta he i dati vengono passati allo strato sottostante, **informazioni di**

controllo specifiche vengono aggiunte .

Quando i dati raggiungono lo **strato 1**, essi vengono iniettati nel **canale fisico** che connette i due hosts.

Quando i dati raggiungono la destinazione, essi attraversano i layers **dal basso verso l'alto** in ordine, per raggiungere **lo strato n**.

Tra ogni **layer adiacente** viene **definita un'interfaccia**; essa specifica le **operazioni primitive** che possono essere richieste da un **altro layer adiacente** per offrire servizi al layer chiamante.

Accedere ai servizi di strato

Un layer **n** utilizza i servizi dello strato **n-1** per aggiungere nuove funzioni e per fornire un servizio **aggiuntivo** allo strato **n+1**; in parole povere lo strato n prende dal suo sottostante dei servizi, ed aggiunge delle funzioni per lo strato soprastante.

I servizi di uno strato vengono acceduti attraverso un **SAP - Service Access Point**; ogni SAP ha un **indirizzo:**

L' informazione passata attraverso gli strati è chiamata **PDU - Protocol Data Unit**. Per uno specifico layer **n** è chiamato **n-PDU**. Quando un **n-PDU** entra in uno strato è chiamato **SDU - Service Data Unit**.

Un **PCI - Protocolo Control Information** è aggiunto ad esso ed quindi diventa un **(n-1-PDU)** da passare allo strato n-2 (non ci ho capito niente sinceramente).

Comunicazione tra layers

In ogni architettura, la comunicazione tra layers segue lo schema:

1. L'Host1 invia un messaggio **M** dall' **application layer** (5) dell'Host1 allo stesso strato dell'Host2.
2. Lo strato 4 aggiunge uno specifico **header** al messaggio **M** e passa il risultato allo strato 3.
3. Lo strato 3 **frammenta** il messaggio ed aggiunge ad ogni frammento uno specifico **header**.
4. Lo strato 2 aggiunge un **trailer** ed inietta i dati all'interno del **canale fisico**.
5. I dati vengono trasferiti all'Host2.
6. Sull'Host2, i dati seguono il percorso inverso.

Come sono progettati gli strati?

Progettazione degli strati

L'implementazione di ogni strato dipende da un insieme di **decisioni di progettazione** relative a diversi problemi:

- Identificazione del **sender** e del **receiver** di una conversazione; noto come **addressing**.
- Meccanismo per selezionare percorsi alternativi (**routing**) per attraversare routers.
- Meccanismo per preservare, o ricostruire, **l'ordine originale** dei dati.
- Tecniche per costruire diversi **canali virtuali** al di sopra di un **canale fisico**. Questa tecnica è nota come **multiplexing**. Permette a diverse conversazioni di **coesistere**.

Dobbiamo anche avere una **gestione degli errori**:

- **Detection**

- **Correction**

Dobbiamo decidere la **grandezza del messaggio** (minima o massima), e la possibile **frammentazione**.

Trasferimento dati

- Singola direzione - **simplex communication**
- In entrambe le direzioni ma alternativamente (non in contemporanea) - **half-duplex communication**
- In entrambe le direzioni contemporaneamente - **full-duplex communication**

---

fine lezione 2