



**Università degli Studi di Bologna
Facoltà di Ingegneria**

Corso di
Reti di Calcolatori L-A
Open System Interconnection

Antonio Corradi

Anno accademico 2009/2010

Open System Interconnection

OSI come standard di comunicazione tra sistemi aperti,

OBBIETTIVO: INTEROPERABILITÀ

potere comunicare ed operare tra sistemi eterogenei

ma non solo

OSI standard per la Gestione dei sistemi

(Systems management Network management)

**Come si possono controllare, coordinare, monitorare
sistemi interconnessi eterogenei**

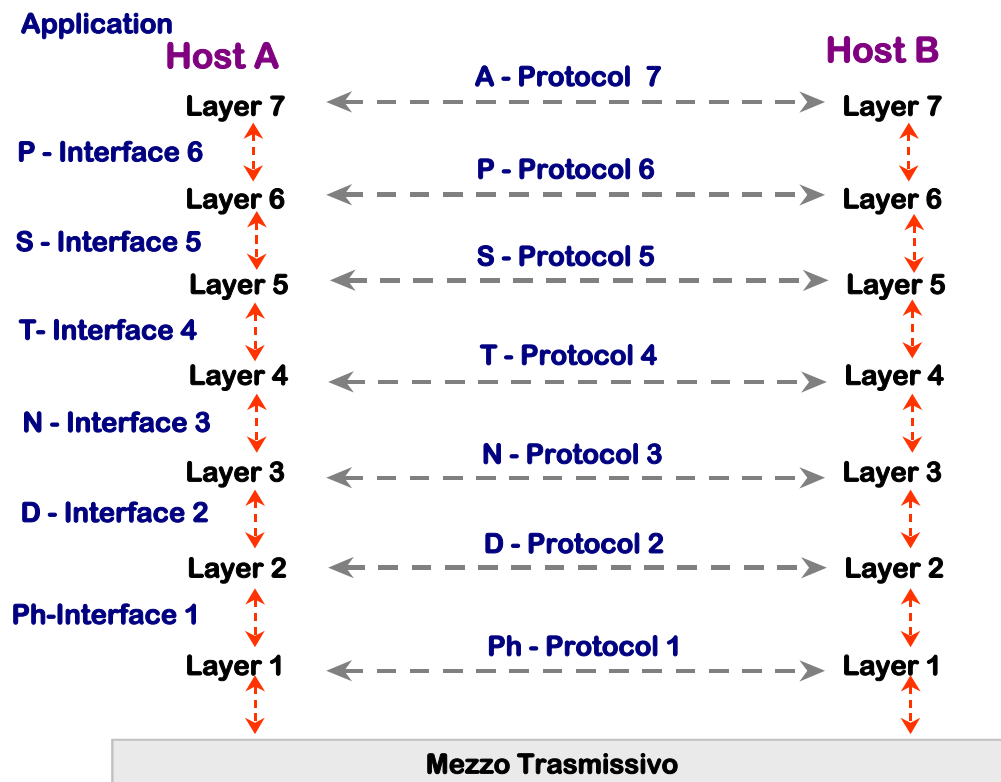
OSI standard e schema di progetto

- **organizzato a livelli precisi e specificati**
- **ad oggetti**
- **come scenario ampio di soluzione**
- **senza legami con realizzazioni (proprietarie o meno)**

OSI A LIVELLI

OSI come insieme di livelli di progetto

Applicazione
Presentazione
Sessione
Trasporto
Network
Data Link
Fisico



per la comunicazione tra pari

OSI A LIVELLI

Partendo dalla applicazione

Ogni livello ha l'obiettivo di **comunicare con il pari** e lo realizza tramite il **protocollo**, usando il servizio sottostante

implementazione

specificata del protocollo

protocollo realizzato dal livello e non visibile al disopra

Ogni livello **fornisce un servizio specificato e disponibile al livello superiore**

definizione di un servizio

semantica del servizio

interfaccia del servizio usata dal livello sovrastante

Ogni sistema a livelli si basa sul principio della **separazione** dei compiti (delega) e della **trasparenza** della realizzazione (astrazione)

OSI applicato in modo estensivo nelle specifiche internazionali e dagli enti di comunicazione

LIVELLI OSI

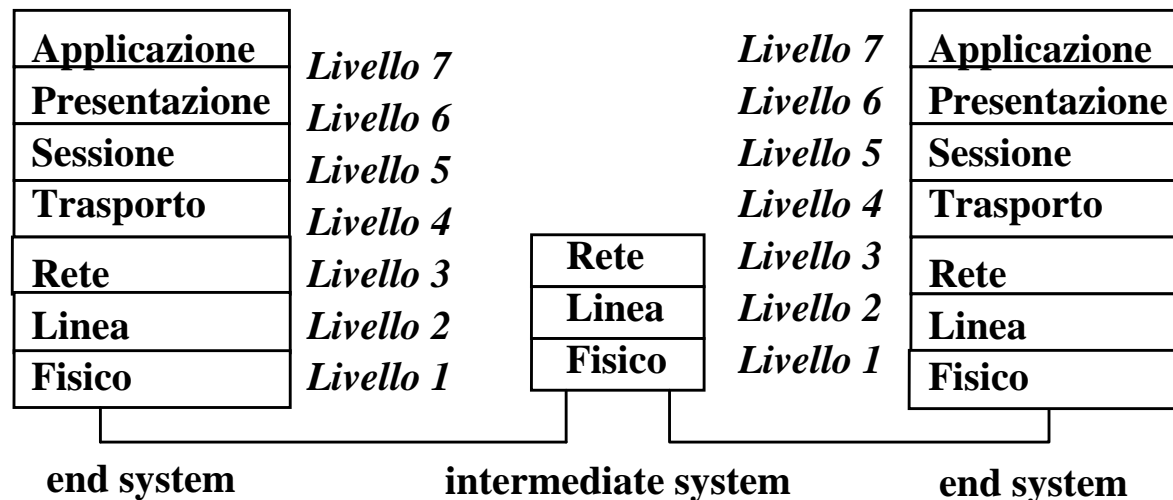
- **definizione del servizio (descrizione verticale)**

definizione astratta dei servizi del livello corrente disponibili al livello immediatamente superiore

in altre parole, l'interfaccia offerta dal livello stesso per l'accesso

- **specifica del protocollo (descrizione orizzontale)**

specifica dettagliata di come il livello fornisce il servizio tramite scambio di dati ed informazioni tra le due realizzazioni dei sistemi comunicanti, ossia la realizzazione del servizio



NOMI ed ENTITÀ IN OSI

Si considerano macchine **host** e anche **nodi intermedi** di rete

Ogni elemento attivo in un livello è detto **entità**

SISTEMI DI NOMI per le entità

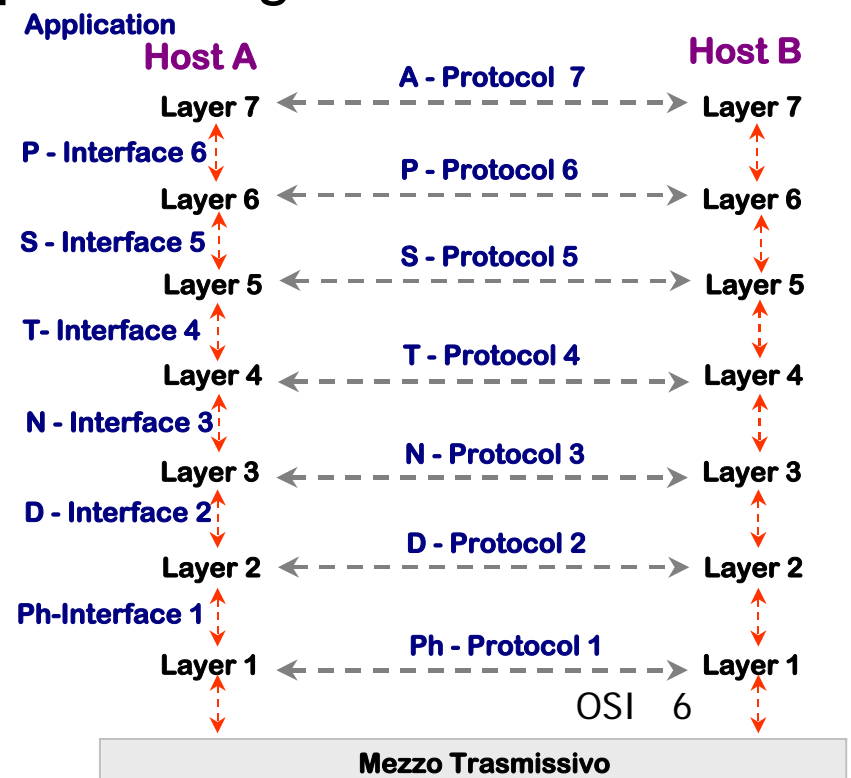
usando la lettera iniziale maiuscola per distinguere

(N)-layer e (N)-protocol

ad esempio:

S-protocol per il livello di Sessione

I nomi non sono usati solo per
i livelli e protocolli



OSI - SAP

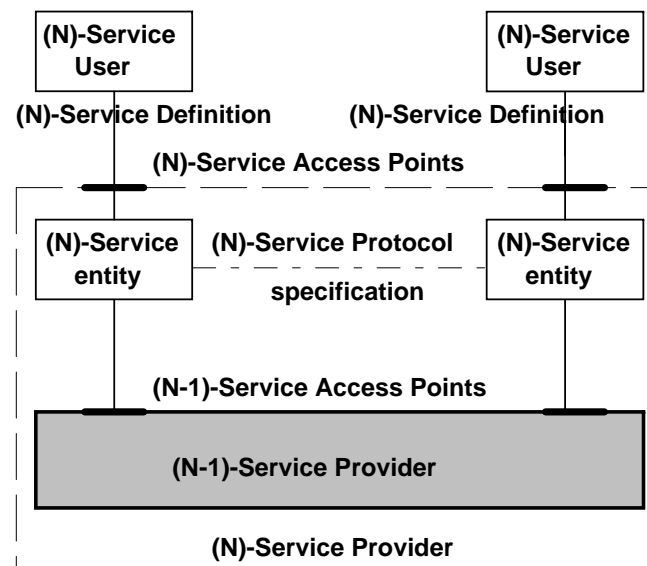
SERVICE ACCESS POINT o (N)-SAP

Interfaccia logica tra una (N-1)-entity ed una (N)-entity

API le funzionalità disponibili ad un SAP

un (N)-address identifica un insieme di SAP al confine tra il livello (N) ed il livello (N+1)

(N)-SAP address un indirizzo unico che identifica un singolo (N)-SAP



OSI – NOMI e SAP

Per le entità OSI

NOMI UNICI

Ogni SAP deve avere un nome unico per essere identificata e ritrovata

NOMI delle ENTITÀ

per identificare una entità si devono nominare tutti i SAP di ogni livello fino al livello 1

Scorciatoia: si parte con i nomi di rete che identificano il nodo in modo unico

Una entità (un pari) viene identificato come **SAP di rete** e con la sequenza a stack dei **nomi** delle **entità** per identificare il livello superiore (ad esempio: IP, porte, socket, processi, ...)

In questo modo abbiamo la possibilità di specificare in modo non ambiguo ogni partecipante

OSI – AZIONI di COMUNICAZIONE

In genere, per una azione

Mittente

Entità che ha la responsabilità di iniziare la comunicazione

Ricevente

Entità che accetta la comunicazione e poi la sostiene

Intermediari

Eventuali intermedi che devono partecipare alla comunicazione intesa come risorse per sostenerla

Il mittente manda dei dati ad un ricevente che può anche rispondere all'invio con un'azione applicativa conseguente

Ogni azione comporta una comunicazione che passa attraverso i livelli da applicativo a fisico, del mittente e ricevente e almeno fino al livello di rete per gli intermediari

OSI – COMUNICAZIONE

Per la comunicazione tra i diversi livelli si riconoscono

- IDU (INTERFACE DATA UNIT)
- PDU (PROTOCOL DATA UNIT)

che descrivono i messaggi che permettono di **chiedere il servizio** e di **specificare il protocollo**

IDU della sessione, con cui si chiedono i servizi alla S-SAP

PDU del trasporto, con cui si mandano messaggi tra pari T

Le informazioni che vanno verso il basso sono incapsulate e devono poi essere trattate dal protocollo, per capire come agire

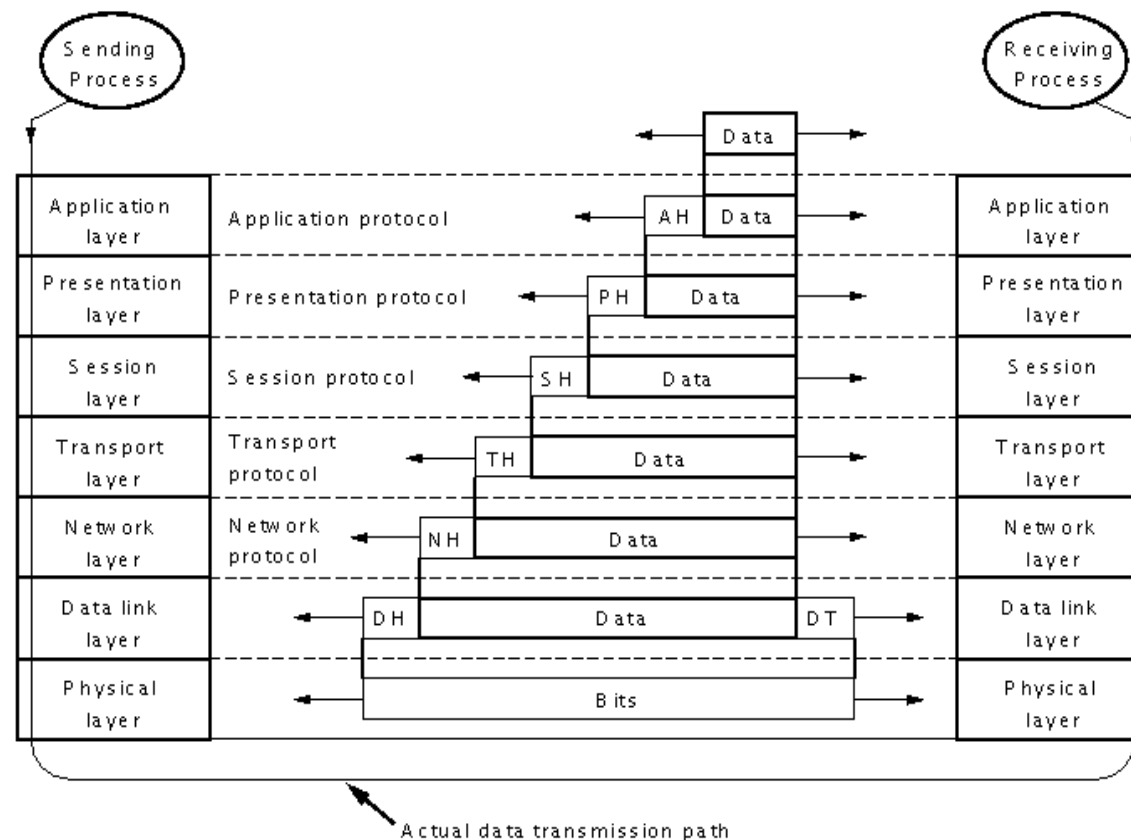
All'interfaccia con il livello sottostante IDU contiene anche la parte di specifica di protocollo: **ICI (Interface Control Information)** coordina operazioni, determinando il protocollo e definendo il PCI

PCI Protocol Control Information

Il PDU viene formato aggiungendo informazioni al dato passato

OSI – COMUNICAZIONE

Su iniziativa del mittente, ogni livello introduce le proprie specifiche a quelle del dato e le passa al SAP sottostante, ... fino al livello fisico, dopo il quale si comincia a risalire fino all'Application corrispondente



OSI – PROTOCOLLO

OSI definisce le **sole specifiche di comunicazione** senza dettare nessuna specifica a livello locale né suggerire alcuna tecnologia di soluzione

Ad esempio, non si dice mai come i livelli di protocollo devono essere realizzati, a processi, procedure, ad oggetti, ecc

Anzi, si evita qualunque ricopertura con termini che suggeriscano un soluzione

La organizzazione a SAP è astratta e potrebbe essere applicata alla modellizzazione anche di molti altri sistemi.

Per ogni livello, sono possibili e riconosciute:

- implementazioni a procedure
- implementazioni a processi
- ancora più parallele

Non si parla mai di processi, procedure, ma si usano termini come attività

OSI – MODALITÀ

CONNECTIONLESS. Ogni unità di dati è trasferita in modo indipendente dalle altre unità ed è autocontenuta (senza ordine)

Nessuna **qualità del servizio** e nessuna negoziazione e valutazione

Lo scambio di informazioni tra i due pari avviene **senza storia e senza nessun concetto di negoziazione**

CONNECTION-ORIENTED. Si stabilisce una connessione tra entità pari che devono comunicare, con caratteristiche della connessione negoziate durante la fase iniziale

In modalità connection-oriented la comunicazione tra due utenti di pari livello avviene in tre fasi:

1. apertura della connessione
2. trasferimento di dati sulla connessione
3. terminazione della connessione

Il servizio connection-oriented di un livello deve fornire le opportune funzionalità per le tre fasi con la richiesta **qualità del servizio**

OSI – MODALITÀ

CONNECTIONLESS

adatta per **dati occasionali** e **senza qualità** della comunicazione

Un messaggio mandato dopo può arrivare prima di un precedente

Costo limitato ma scarse garanzie

CONNECTION-ORIENTED

Il pari con iniziativa deve prima provvedere alla connessione che spesso porta anche a stabilire quali sono gli intermediari per la connessione stessa

la comunicazione può avere luogo in modo bidirezionale sulla connessione e si ottiene qualità, come l'ordinamento dei messaggi, il coordinamento delle risorse, ...

Su una connessione costi più elevati, ma maggiori garanzie

La connessione non significa impegno di risorse su eventuali nodi intermedi necessariamente (vedi Internet)

OSI – QUALITÀ

OSI considera il servizio come caratterizzato da attributi che costituiscono la **Qualità di Servizio (QoS)** e permette la scelta

Ogni servizio deve fare i conti con la qualità logicamente richiesta e le possibilità reali di risorse richieste e disponibili

Proprietà ed esempi di Servizi

affidabilità della comunicazione e **sequenza** dei flussi di dati

connessione	affidabile	non affidabile
--------------------	-------------------	-----------------------

non connessione	affidabile	non affidabile
------------------------	-------------------	-----------------------

***datagramma** senza connessione non affidabile*

***connessioni affidabili** sequenze di messaggi*

Ma anche richieste diverse: solo garanzia di sequenza di dati che si possono anche perdere, anche dati che vogliamo che siano ricevuti, sequenze di byte che richiediamo ricevute in modo unico,

...

OSI – PRIMITIVE

Le due entità pari cooperano tramite primitive per implementare le funzionalità del livello cui appartengono

Primitive base fornite sono dette: **data** per trasmettere contenuto e **connect**, **disconnect** per aprire chiudere la connessione

Quattro possibili forme per una primitiva

- **Request** ⇒ il service user richiede un servizio (una azione)
- **Indication** ⇒ il service provider indica al service user che è stato richiesto un servizio (segnalazione di evento)
- **Response** ⇒ il service user specifica la risposta alla richiesta di servizio (una azione)
- **Confirm** ⇒ il service provider segnala la risposta alla richiesta di servizio (segnalazione di evento)

POSSONO ESSERE PRESENTI ANCHE TUTTE LE FORME (nel caso sincrono)

OSI – FORMA delle PRIMITIVE

Nel dialogo tra pari si possono utilizzare le forme

S-CONNECT.response Sintassi: **Nome primitiva** **punto** **Tipo primitiva**

Primitiva asincrona

nessuna conferma al cliente

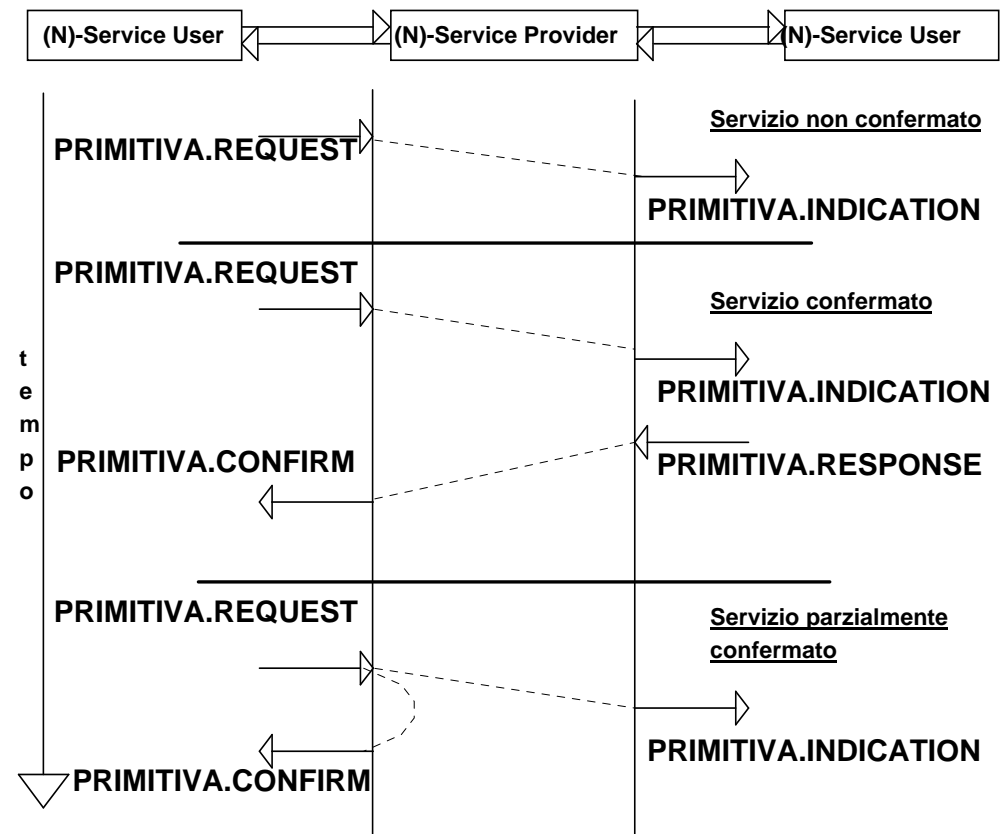
Primitiva sincrona

(risultato al cliente) con conferma
e azione al servitore

Primitiva asincrona bloccante

solo conferma al cliente

Si noti l'evento non richiesto dall'utente
stimolato dalla comunicazione (la parte indication)



SEQUENZA delle PRIMITIVE

Tipica sequenza con connessione attivata dall'iniziatore

CONNECT	apertura della connessione con negoziazione
molte DATA	invio dati
DISCONNECT	chiusura della connessione

Per la CONNECT

CONNECT.request - Richiesta di stabilire una connessione

CONNECT.indication - segnale al chiamato

CONNECT.response - Il chiamato accetta/rifiuta di connessione

CONNECT.confirm - Conferma al chiamante l'avvenuta connessione

Per la trasmissione dei messaggi DATA

DATA.request - Invio dati

DATA.indication - Segnala dati con la qualità della connessione

OSI vs TCP/IP INTERNET

connessione OSI tipicamente con **QoS** e **impegno intermedi**

connessione TCP/IP solo **best effort** e impegno **solo endpoint**

3 Livelli inferiori OSI Fisico, Data Link, Rete

1 Livello intermedio Trasporto

forniscono un meccanismo trasparente per il trasporto end-to-end

le funzioni base dei quattro livelli includono:

- controllo degli errori dovuti al rumore o altra causa
- controllo di flusso dei dati
- modelli di indirizzamento per identificare end system (naming)
- per rete le strategie di routing per trasferire i dati (internetworking)

Protocolli

livello fisico ripetitore protocollo RS232

livello data link bridge protocolli Ethernet, HDLC, PPP

OSI - LIVELLO NETWORK

Livello di RETE tiene conto dei nodi intermedi tra due pari

Impossibilità di controllare direttamente il cammino da un qualunque mittente ad un qualunque destinatario

Il livello di network si occupa delle diverse realizzazioni di routing tra reti diverse oltre a definire il sistema di nomi delle entità

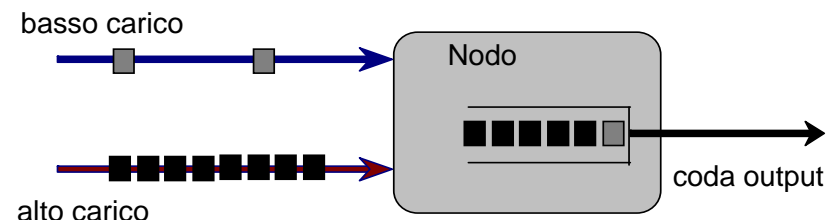
Obiettivo: passaggio delle informazioni interferendo meno possibile sul comportamento locale

COMPITI del LIVELLO di RETE

- Indirizzamento (vedi nomi di IP)
- Controllo di flusso tra due pari
- Controllo di congestione nel sistema intero

OBIETTIVI: migliorare efficienza ed evitare ingiustizia, deadlock

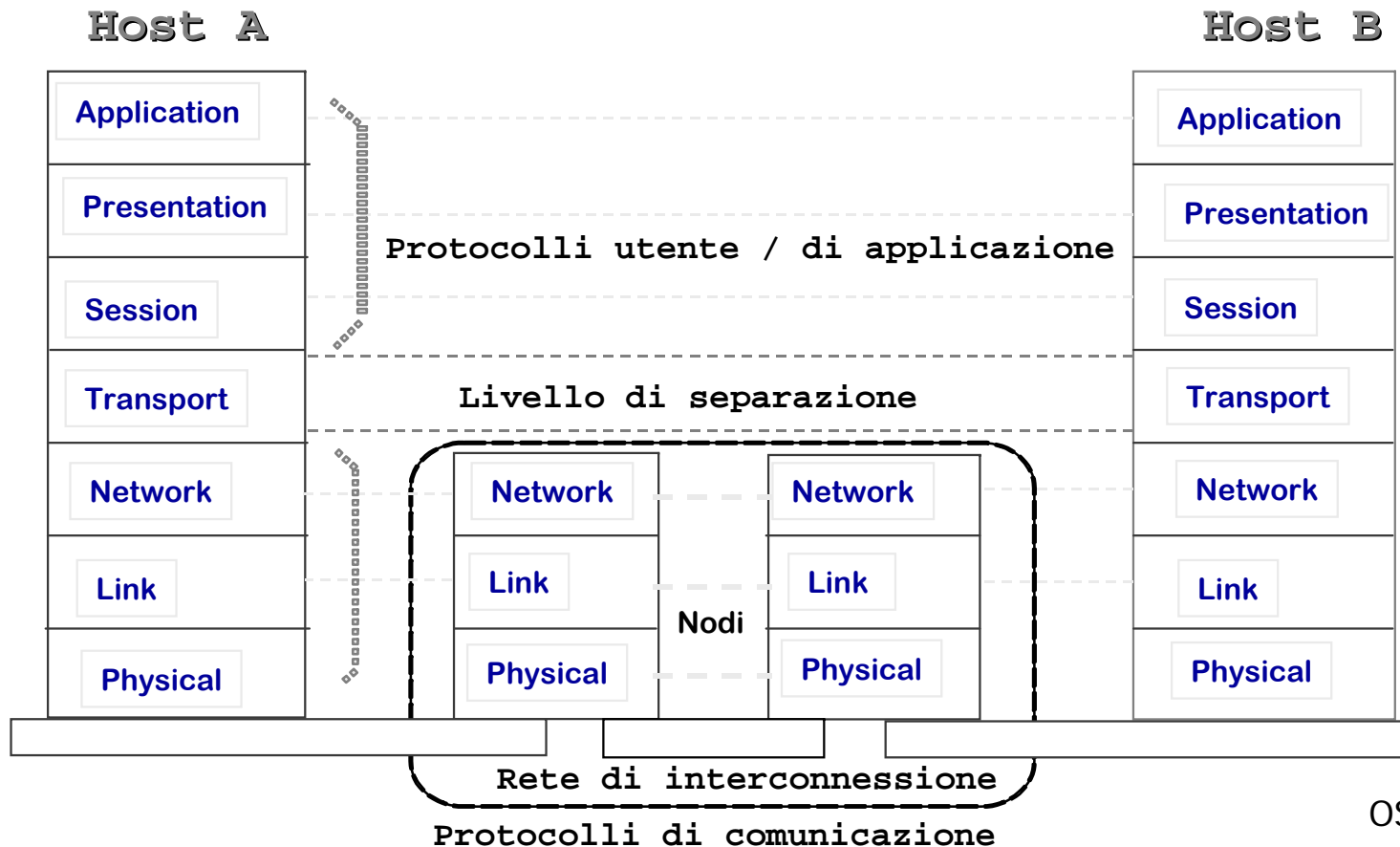
Principio di separazione: i nodi intermedi devono potere interagire solo per le funzionalità necessarie e non essere toccati ai livelli applicativi



OSI - LIVELLI e TRASPORTO

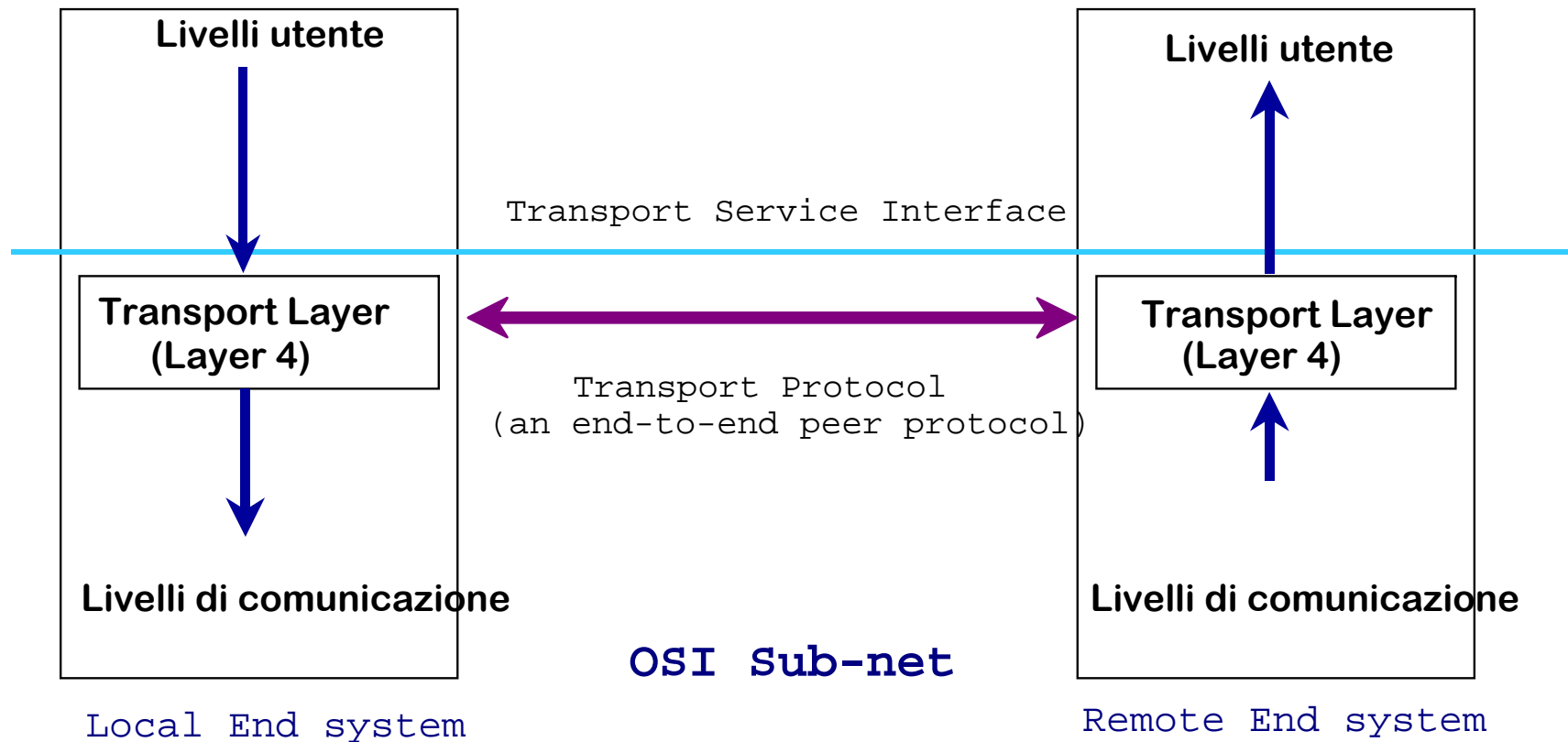
Comunicazione
Applicazione

i livelli fino sotto il trasporto
i livelli sopra il trasporto



OSI - LIVELLI e TRASPORTO

il trasporto separa i livelli applicativi da quelli fisici
Applicazione si localizza sopra al trasporto



OSI - LIVELLI e TRASPORTO

il Trasporto comincia a considerare la struttura dei nodi partecipanti, in particolare gli endpoint

Ogni livello ha le proprie entità e SAP

Se focalizziamo solo T e R, un nodo potrebbe avere molte SAP di trasporto e una sola di rete: in questo caso le applicazioni devono specificare quali enti sono coinvolti per ogni comunicazione

Lo stesso discorso per ogni livello superiore

Un pari che vuole comunicare deve specificare tutta la pila di SAP proprie e anche quelle che permettono di arrivare all'altro in modo non ambiguo

Anche gli intermedi devono essere considerati per la connessione con qualità e devono essere negoziati

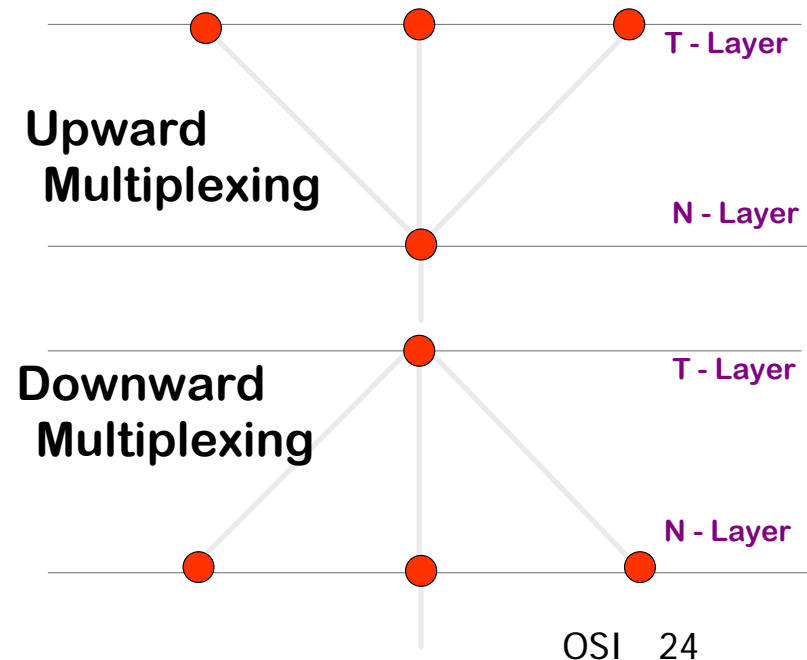
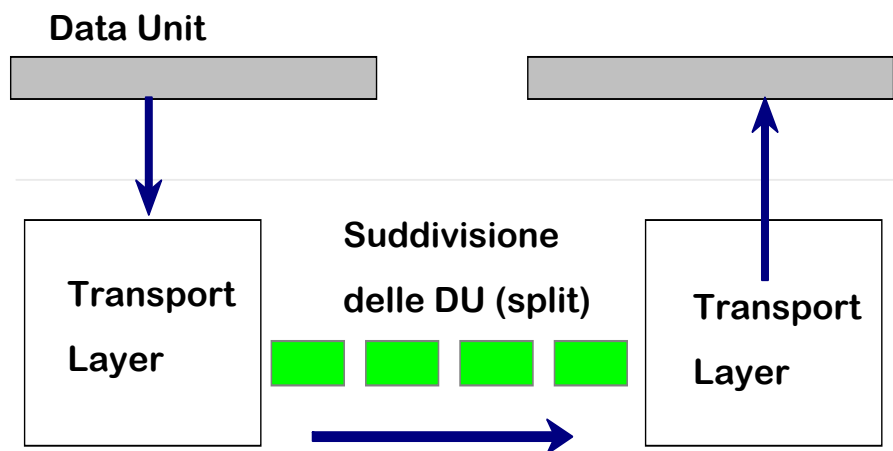
La primitiva CONNECT mette in gioco molte entità su tutti i nodi interessati alla connessione e mantiene le risorse impegnate

OSI – LIVELLO di TRASPORTO

Livello T - Funzioni possibili

Il trasporto può **spezzare** il dato e **ricomporlo** dopo averlo portato suddiviso fino al pari

Il trasporto può lavorare **unendo** o **decomponendo** flussi di trasporto rispetto a quelli di rete (**Multiplexing**)



LIVELLO TRASPORTO

Trasporto come **livello end-to-end** per **separare i livelli** relativi alla comunicazione da quelli più vicini alla applicazione

Obiettivo - spedizione di dati sul canale di connessione con correttezza, con certi tempi di risposta, e con una certa qualità di servizio, su richieste dal livello S superiore

MODALITÀ CONNECTION ORIENTED

- apertura e terminazione di una connessione
- trasferimento di dati normali e privilegiati (expedited)

I dati expedited sono soggetti ad un controllo di flusso separato che permette l'invio di messaggi di controllo anche se il servizio per i dati normali è bloccato

In generale, ogni primitiva di servizio per un dato livello prevede un certo insieme di parametri

Anche MODI NON CONNESSI (vedi INTERNET)

PRIMITIVE LIVELLO TRASPORTO

Livello di Trasporto agisce su base end-to-end

Primitive del servizio di Trasporto semplici e con relativamente pochi parametri significativi

<u>primitiva</u>	<u>tipo di servizio</u>	<u>parametri di servizio</u>
T-CONNECT	servizio confermato	indirizzo del chiamante e del chiamato, opzione per l'uso di dati privilegiati, qualità di servizio e dati d'utente.
T-DATA	servizio non confermato	dati di utente
T-EXPEDITED-DATA	servizio non confermato	dati di utente
T-DISCONNECT	servizio non confermato	ragione della terminazione, dati d'utente

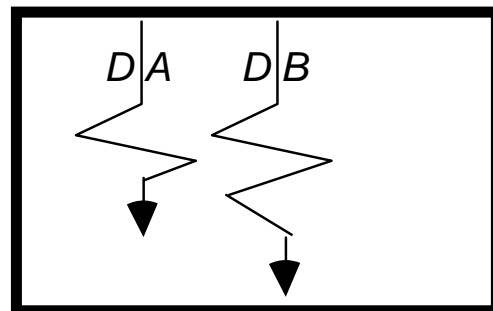
LIVELLO SESSIONE

Se il trasporto lavora da nodo a nodo, la prima esigenza della sessione è il supporto al dialogo

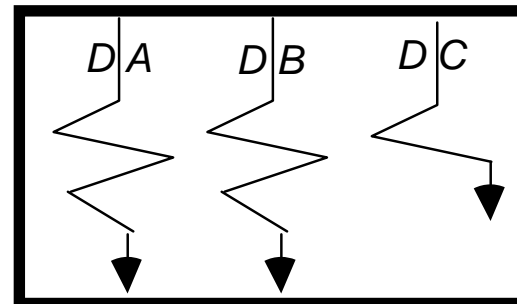
LA SESSIONE considera e determina i meccanismi per il dialogo tra entità tenendo in conto le possibilità tra due pari che comunicano

Nella sessione, il dialogo può

- essere bidirezionale
- essere molteplice e strutturato in attività separate e diverse
- considerare le risorse impegnate
- avere garanzie di correttezza e affidabilità



Applic A



Applic B

PRIMITIVE LIVELLO SESSIONE

Il livello di Sessione coordina il dialogo basandosi sul servizio offerto organizzato in **unità funzionali**, ognuna legata ad un insieme di primitive e parametri

Il numero delle unità funzionali cresce per i livelli verso l'applicazione

Servizio di Sessione offre **58 primitive** raggruppate in **diverse unità funzionali**

uso di connessione, con QoS negoziata, e semantica più complessa

DIALOGO strutturato attraverso

- azioni di controllo, come trasferimento di iniziativa, ecc
- attività: il dialogo è diviso in attività indipendenti che si possono gestire (iniziare, sospendere, cancellare, ecc.)
- eccezioni: è possibile notificare eccezioni al servizio corrispondente

Ogni pari può richiedere il livello di servizio adatto alle esigenze sue e della invocazione

SERVIZI LIVELLO SESSIONE

Il **livello di Sessione** offre servizi analoghi a quelli del livello di Trasporto, ed ogni livello:

- **apertura della connessione e sua terminazione**
- **trasferimento dati**

si possono avere fino a quattro tipi di dato

Servizi aggiunti e specializzati per:

- **gestione dell'interazione**
modalità di dialogo half-duplex, full-duplex o simplex
- **sincronizzazione**

inserimento dei punti di sincronizzazione (checkpoint) e
gestione delle eccezioni

I punti di **sincronizzazione sono nuovi dati**, come le **autorizzazioni** messe in gioco negli stati della comunicazione

SINCRONIZZAZIONE SESSIONE

Possibilità di intervenire sul dialogo tra pari

Trasmissione di un file da due ore bloccata dopo un'ora ⇒

Si riprende dal risultato del trasferimento precedente

Se si verificano errori nella comunicazione ⇒ roll-back

Nel trasferimento di molti MByte, se crash ⇒ si ricomincia (?)

Due tipi di punti di sincronizzazione maggiori e minori

determinati per ritornare ad uno stato definito e concordato dalle due entità di Presentazione (cioè gli SS-user)

1) **punti di sincronizzazione maggiore**

il mittente attende (modo **sincrono bloccante**) che il ricevente confermi

2) **punti di sincronizzazione minore**

invio con conferma o meno, il mittente può continuare a spedire dati o punti di sincronizzazione. Il mittente non deve segnalare la ricezione di un punto minore e la conferma di un punto minore conferma anche tutti i punti precedenti

PUNTI di SINCRONIZZAZIONE

I punti di sincronizzazione usati per ri-sincronizzarsi verso uno stato definito dai punti stessi in caso di recovery

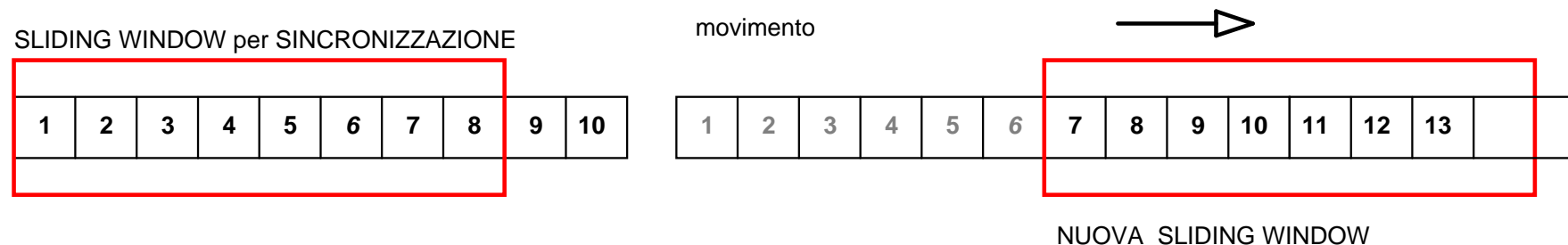
In caso di punti maggiori, attesa

In caso di punti minori, accumulo e poi anche attesa

Tipicamente, si può negoziare il **numero di punti di sincronizzazione minore che possono rimanere in attesa di conferma**

Si determina la dimensione di una **finestra che scorre (sliding window)** di punti non confermati

Al riempimento della finestra, si deve aspettare conferma per procedere



SERVIZI LIVELLO SESSIONE

I punti di sincronizzazione di un dialogo possono essere usati nel recovery per ritrovare uno stato significativo

In caso di recovery, molte strategie diverse

Quelle previste consentono

1) **abbandono**: reset della comunicazione

L'utente può decidere di ripeterla

2) **ripristino**: la comunicazione è riportata nello stato precedente

L'ultimo punto di sincronizzazione maggiore confermato (anche con raffinamenti)

3) **ripristino diretto dall'utente**: la comunicazione è riportata in uno stato arbitrario senza controllo delle conferme "mancanti" di punti di sincronizzazione

È compito delle applicazioni decidere in modo coordinato su uno stato da cui ricominciare la trasmissione

SERVIZI di SESSIONE

Strutturazione e sincronizzazione del dialogo attraverso oggetti astratti detti token

un solo utente possiede il token in ogni momento ed ha il diritto di uso di un insieme di servizi di Sessione

La primitiva **S-CONNECT** per stabilire la connessione permette di negoziare anche i token

ad esempio, l'utente che richiede la connessione e l'utente che la accetta indicano i servizi da implementare

L'intersezione dei due insiemi di requisiti determina la S-connessione

Vari tipi di token distinti come diritto di:

- * **data token**: spedire i dati in Half Duplex
- * **release token**: richiedere la terminazione
- * **synchronize minor token**: creare punto di sincronizzazione minore
- * **synchronize major token**: creare punti maggiori

LIVELLO di PRESENTAZIONE

La codifica delle informazioni non è univoca e ogni pari può usare codifiche diverse

Il livello di Presentazione offre tutti i servizi offerti per trasformare la codifica dei dati ricevuti dai pari e da mandare ai pari

NECESSITÀ di codifiche diverse:

- **differenze naturali tra i sistemi che comunicano**
- **migliorare la comunicazione (efficienza e sicurezza)**
uso di compressione dei dati (efficienza) e crittografia dei dati (sicurezza)

I dati devono essere scambiati dopo un accordo tra i pari che superi gli eventuali problemi di eterogeneità

- | | |
|---------------------------------------|----------------------|
| - linguaggi di programmazione diversi | C, ADA, C#, Java,... |
| - sistemi operativi diversi | UNIX , VMS, WINXX |
| - architetture diverse | ALPHA, RISC, ... |

MOLTI CASI DIVERSI

SE NON ci sono problemi di FORMATO DATI

Cioè tutti usano lo stesso formato e nessuna disomogeneità

⇒ non si fanno trasformazioni vedi in Java

ALTRIMENTI bisogna tenerne conto nel progetto

Cioè molti formati diversi e abbiamo ETEROGENEITÀ?

SE C'È ACCORDO... ⇒ (parte concreta dell'accordo)

trasformazione da un pari ad un altro

vedi stringhe nelle diverse architetture big-endian vs. little-endian

In Internet big-endian

SE NON C'È ACCORDO... ⇒ (parte astratta dell'accordo)

Come si può accordarsi? senza un linguaggio comune

Bisogna determinare un linguaggio comune per l'accordo

DATI in FORMATI ETEROGENEI

In caso di disomogeneità dei dati,

in ogni caso dobbiamo anche chiederci come viaggiano i dati:

- come pura rappresentazione (valori)
- anche con dei descrittori del dato (descrizione e valore)

Solo i valori efficienza, anche descrizione affidabilità

per la comunicazione tra nodi eterogenei due soluzioni:

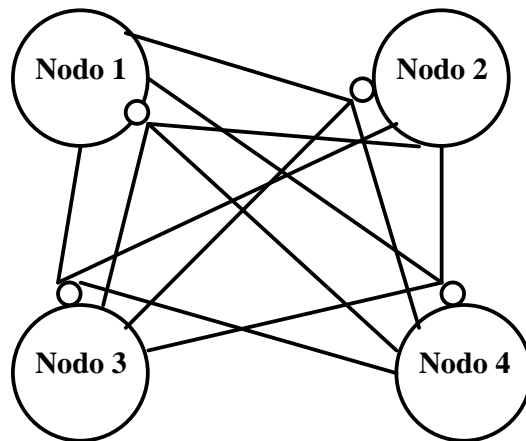
1. dotare ogni nodo di tutte le funzioni di conversione possibili per ogni possibile rappresentazione dei dati
2. concordare un formato comune di rappresentazione dei dati: ogni nodo possiede le funzioni di conversione da/per questo formato

La prima ⇒ **elevata performance**

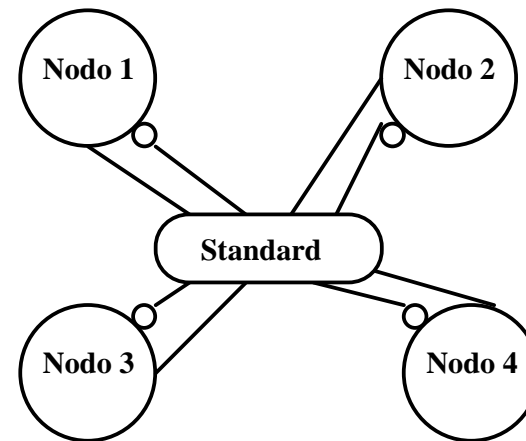
La seconda ⇒ **implementazione di un minore numero di funzioni di conversione**

DATI in FORMATI ETEROGENEI

In caso di N nodi eterogenei
nel primo caso le funzioni di conversione sono $N*(N-1)$
nel secondo sono N verso ed N per il formato comune



Sono necessarie 12 funzioni
di conversione del formato di dati



Sono necessarie 8 funzioni
di conversione del formato di dati

Nel primo caso ogni volta una trasformazione

Nel secondo ogni volta due trasformazioni

Si usa sempre e solo la seconda

ANCORA per l'ACCORDO

SE C'È ACCORDO \Rightarrow TRASFORMAZIONI E RIDONDANZA

Possiamo usare diversi gradi di ridondanza a secondo del costo associato alla comunicazione (impegno di banda)

I dati possono essere solo **valore**
 lunghezza del valore, valore
 tipo, lunghezza campo valore, valore

SE NON C'È ACCORDO... \Rightarrow necessità di accordarsi

Come si può accordarsi senza un linguaggio comune?

Diventa necessario avere un qualche linguaggio o notazione comune e standard per l'accordo

Notiamo (parte astratta dell'accordo)

anche in caso di omogeneità potrebbe essere necessario raggiungere un accordo

ESEMPI di ACCORDO

Uso del linguaggio comune noto a entrambi

In caso di telnet, si è definito un linguaggio ad-hoc per l'accordo sulle specifiche del video standard

Chi fa l'echo?

Lo fai tu?

Io no, Fallo tu!

No, tu! ...

E se non ci fosse un linguaggio comune, si ricorre ad un linguaggio noto ad entrambi per il dialogo di accordo

Cosa mi dici?

File HTML con nome, cognome

preferirei cognome e nome

OK

parliamo dei componenti 1.3.6.1.2.1.4

IL CASO della PRESENTAZIONE

Necessità di accordarsi e definire

- **il contesto di comunicazione**
- **il soggetto della comunicazione**
- **la semantica delle informazioni**
- **le informazioni vere e proprie**

Il contesto potrebbe definire di cosa stiamo parlando (magazzino), il soggetto di cosa (oggetti parti), per poi dare significato (tipo di attributi) e specifica di ciascuno

Il livello di presentazione stabilisce come negoziare e definire una base comune

Spesso l'accordo può essere ottenuto solo con protocolli detti di negoziazione, ossia a molte fasi (con durata anche non predefinita e anche lunga nei casi peggiori)

PROTOCOLLI di NEGOZIAZIONE

Protocolli con numero di fasi non predicibile spesso determinato da eventi verificatisi durante il protocollo stesso

BIDDING (Contract Net) tra sender e receiver si prevedono molte fasi, almeno 5 anche ripetute

- 1) il sender fa un broadcast della propria esigenza
- 2) i receiver fanno un'offerta (bid)
- 3) il sender sceglie tra i bid dei receiver
- 4) il receiver accoglie l'ok definitivo (contract)
- 5) accordo

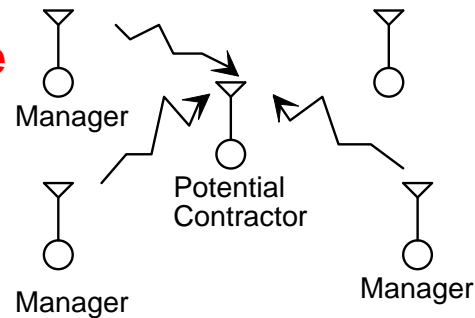
Non ci sono prenotazioni: alla fase 4, il receiver può rifiutare e si riparte da 3 (o peggio da 1)

SELEZIONE molto FLESSIBILE ma COSTOSA

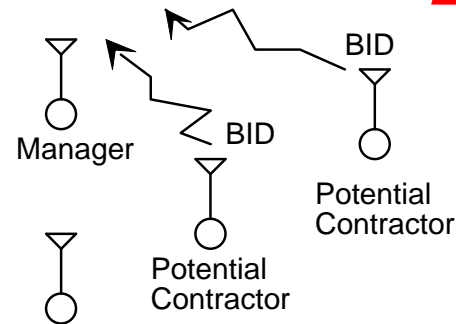
BIDDING (Contract Net)

Protocolli a fasi multiple

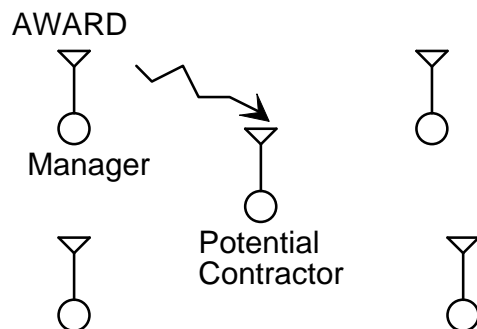
1) Announce richiesta



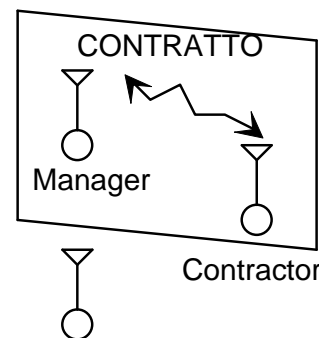
2) Fase di Bidding



3) Scelta del Bidder



4) Contratto finale



PROTOCOLLI di PRESENTAZIONE

Il livello di presentazione definisce:

- **un linguaggio astratto di specifica** (parte astratta accordo)
(ASN.1 Abstract Syntax Notation) per casi estremi
 - **e uno concreto di descrizione dei dati** (parte concreta accordo)
(BER Basic Encoding Rules) sempre usato
- definendo un **contesto di comunicazione**
 - distinguendo informazioni in forma **astratta e concreta**
 - permettendo di specificare i dati sia **in modo astratto e le informazioni di controllo** in modo indipendente dalla forma concreta (se è il caso, e solo se necessario)
 - consentendo di definire la **forma comune di rappresentazione concreta dei dati** (non coincidente con la precedente astratta e richiedendo le usuali trasformazioni da forma locale a forma comune)

BER - Linguaggi di PRESENTAZIONE

BER - sempre necessario

(BER Basic Encoding Rules)

sempre usato

Triple Tag-Length-Value:

codifica a discesa ricorsiva

```
address.source = "Suna"  
address.destination = "Decb"  
length = 3  
data = 'x', 'y', 'z'
```



<u>Primitive types</u>	<u>BER</u>
BOOLEAN	
INTEGER	02
OCTETSTRING	04
IA5String	16
<u>Constructor types</u>	
SEQUENCE	30
SEQUENCE OF	
SET	
SET OF	
CHOICE	

```
30 22  
  30 12  
    16 04 'S' 'u' 'n' 'a'  
    16 04 'D' 'e' 'c' 'b'  
  02 01 03  
04 03 01 02 03
```

ASN.1- Linguaggi di PRESENTAZIONE

ASN.1 - usato in caso di bisogno (ASN.1 Abstract Syntax Notation) in necessità di accordo

Primitive types

BOOLEAN
INTEGER
OCTETSTRING
IA5String

```
Address ::= SEQUENCE {  
    addr_src    IA5String,  
    addr_dst    IA5String  
}
```

Constructor types

SEQUENCE
SEQUENCE OF
SET
SET OF
CHOICE

```
Pdu ::= SEQUENCE {  
    pdu_ad      Address,  
    pdu_len     INTEGER,  
    pdu_data    OCTETSTRING (SIZE 1024)  
}
```

In molti casi, si possono risparmiare informazioni
(confrontare con XML e HTML: si usa sempre la rappresentazione
dei dati XML con il contenuto HTML?)

LIVELLO di APPLICAZIONE

Il livello di Applicazione è il livello che si interfaccia con l'utente finale della comunicazione in base al modello OSI

Obiettivo **Astrazione**

nascondere la complessità dei livelli sottostanti coordinando il dialogo tra le applicazioni distribuite

Il livello applicativo OSI standard definisce un **insieme di servizi indipendenti dal sistema** e li fornisce a programmi di utente o ad utenti - diversi servizi ed ambienti standard (ISO 9545):

Message Handling System	MHS
Directory service	X.500
System Management	X.700
Common Management Information Service & Protocol	CMISE & CMIP
File Transfer, Access and Management	FTAM
Virtual Terminal Standard	VT
Distributed Transaction Processing	DTP

LIVELLO di APPLICAZIONE

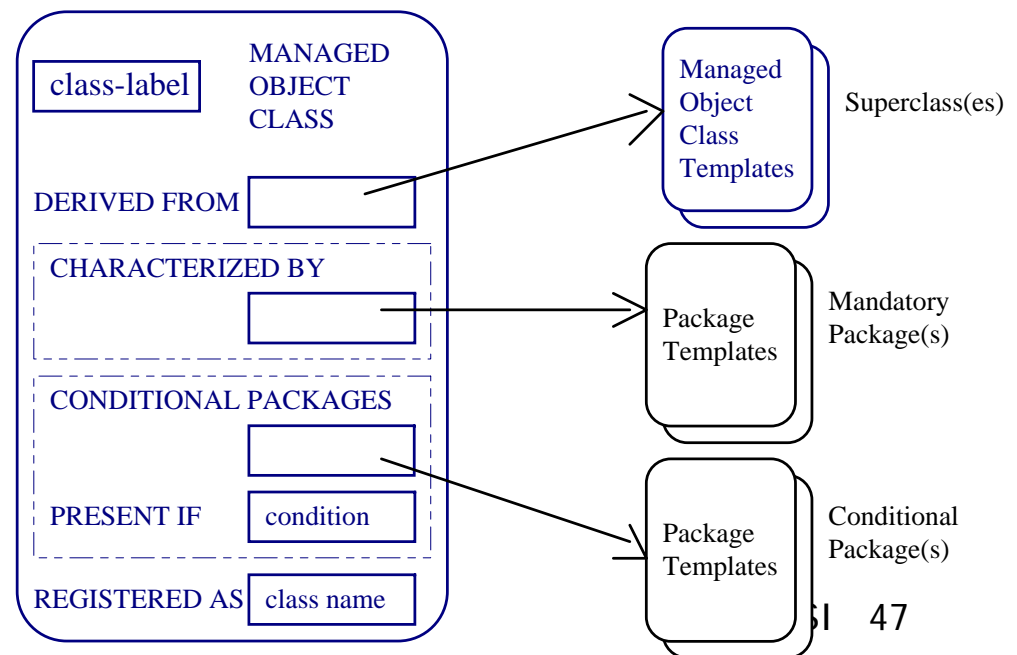
OSI adotta un approccio particolare basato **sul modello ad Oggetti per la specifica delle applicazioni**

Uso di template e package per definire gli oggetti

Pura **ereditarietà statica** tra astrazioni

Oggetti da manipolare come interfaccia ed espressi attraverso l'uso di **package** (anche condizionali)

Si noti la **unicità dei nomi** come presupposto di base
NOMI UNICI come servizio (X.500)

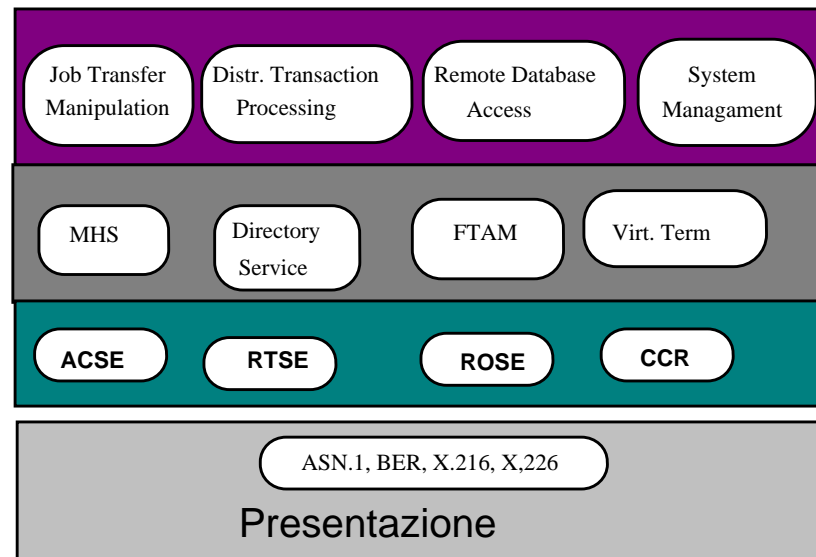


APPLICAZIONE a LIVELLI

Applicazione come insieme di livelli e di strumenti

Altre applicazioni

Applicazione



Alcuni strumenti sono a livello di base rispetto agli altri

ACSE (Association Control Service Element) di base per ogni servizio

RTSE (Reliable Transfer Service Element) per ottenere servizi affidabili

ROSE (Remote Operation Service Element) per operazioni remote

CCR (Commitment Concurrency and Recovery) per azioni multiple coordinate

LIVELLO di NOMI STANDARD

Servizi X.500

Il servizio di direttorio consente di **collocare e classificare ogni entità di interesse** (ogni dispositivo noto) in base al **contenuto degli attributi in un sistema gerarchico di conoscenza molto accessibile (24/7)**

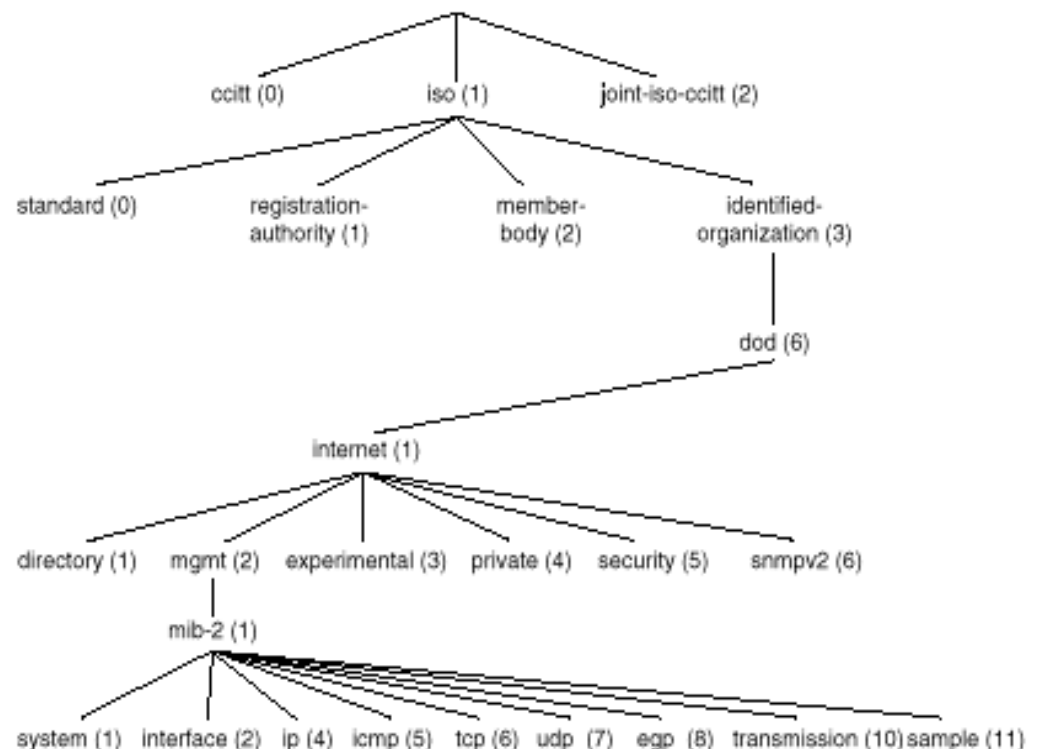
Spesso si riferisce una entità attraverso il suo identificatore specificato come sequenza di decisioni e scelte

1.3.6.1.2.1.4

Il sottoalbero per la descrizione del protocollo IP per gestione

Negli altri casi, si negoziano le proprietà con ASN.1

Per il direttorio, si veda la parte sui sistemi di nomi



OSI e INTERNET

Confronto OSI e TCP/IP, aldilà del numero dei livelli

completezza di OSI
uso di Object-Orientation
interfaccia
implementazione
progetto completo
interesse in standard
definizione ampia e
aperta
Qualità di servizio
Connessione OSI

limiti TCP/IP
descrizione approssimata negli RFC
protocolli e implementazione
insieme
implementazione solamente
prodotti
progetto in crescita

Internet in-segue le specifiche OSI (appena può)