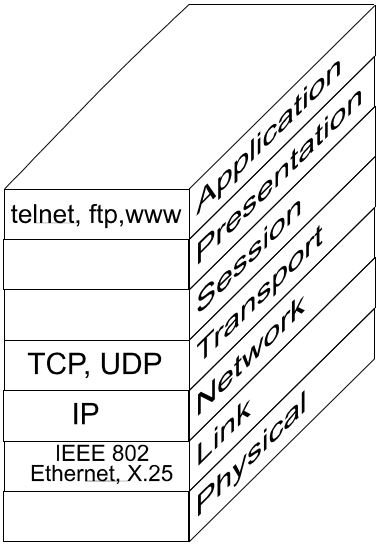
Servizi di UNIX (livello applicazione)



Servizi di tre tipi fondamentali

TERMINALE REMOTO

accesso a nodi remoti

FILE TRANSFER

possibilità di trasferire file tra nodi diversi

COMANDI REMOTI (applicazioni)

esecuzione di comandi remoti, anche specializzati, e riferimenti a servizi remoti
NEWS, MAIL, gopher, WWW (Trasparenza allocazione)

Alcuni sono solo per sistemi UNIX ⇒ **rlogin**, **rwho**, **rsh**, **rup**, ...
Altri più generali ⇒ **ftp**, **telnet**, **mail**, ...

Proprietà fondamentali

Trasparenza (o meno)
Modelli Cliente/Servitore (evoluzioni)
Standardizzazione (evoluzioni)

Servizi APPLICATIVI di un SO

a livello di applicazione per sistemi UNIX: protocolli

Virtual Terminal Procotol: **telnet**

File Transfer Procotol: **ftp**

Trivial File Transfer Procotol: **tftp**

Simple Mail Transfer Procotol: **smtp**

Network News System Transfer Protocol: **nntp**

Line Printer Daemon Procotol: **lpd**

Domain Name System: **dns**

Diffusione conoscenza (più o meno con trasparenza): **nntp, gopher, http, ...**

servizi remoti (UNIX BSD): **rsh**, **rwho**, **rlogin, ...**

telnet e rlogin

Per gestire l'eterogeneità di sistemi operativi ed hardware:
Il terminale locale diventa un terminale del sistema remoto

rlogin per i sistemi UNIX (**r**emote **login**)
telnet standard in TCP/IP internet

Protocollo telnet

telnet costruito su TCP/IP
connessione TCP con **server** per accesso remoto
possibilità di aggancio a server qualunque

Caratteristiche:

- **gestione eterogeneità** tramite interfaccia standard (NVT)

- **Client** e **Server** negoziano le opzioni del collegamento
  (es., ASCII a 7 bit o a 8 bit)

- comunicazione **simmetrica**

Client stabilisce una connessione TCP con **Server**
Client accetta i caratteri dall'utente e li manda al Server
contemporaneamente
accetta i caratteri del server
e li visualizza sul terminale d'utente.
Server accetta la richiesta di connessione del Client e inoltra i dati dalla connessione TCP al sistema locale

Esempio di connessione telnet

telnet con nome logico host o indirizzo fisico IP
anche il *numero di porta*

telnet [host [port]]

Esempio:

```
telnet 137.204.57.33  (telnet lia02.deis.unibo.it)
username:beppe
password:*******
```

TERMINALE VIRTUALE

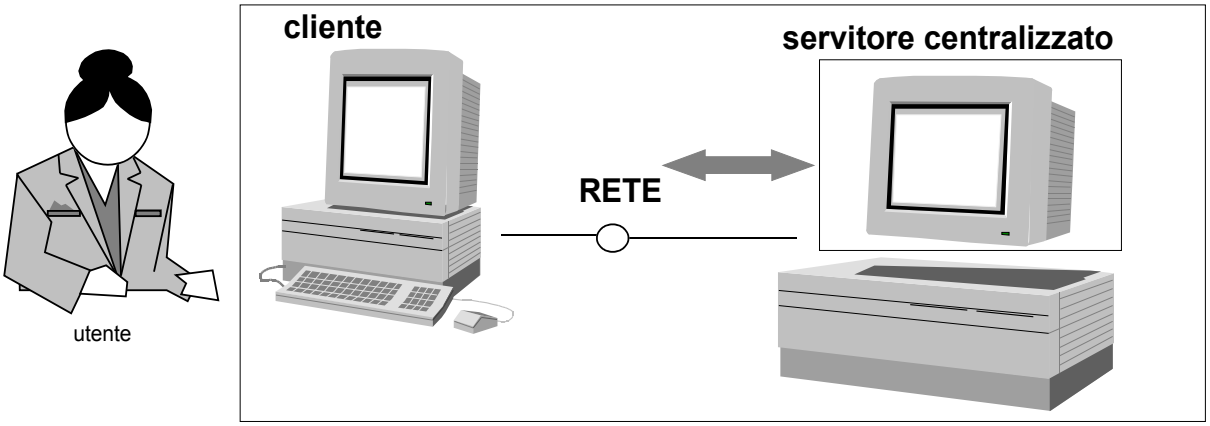
esigenza sentita in generale nelle reti e in particolare nel internetworking

problema: mancanza di standardizzazione dei terminali
I terminali possono differire gli uni dagli altri per:

- il *set* di caratteri
- diversa *codifica* dei caratteri
- la *lunghezza* della linea e della pagina
- i *tasti funzione* individuati da diverse sequenza di caratteri (escape sequence)

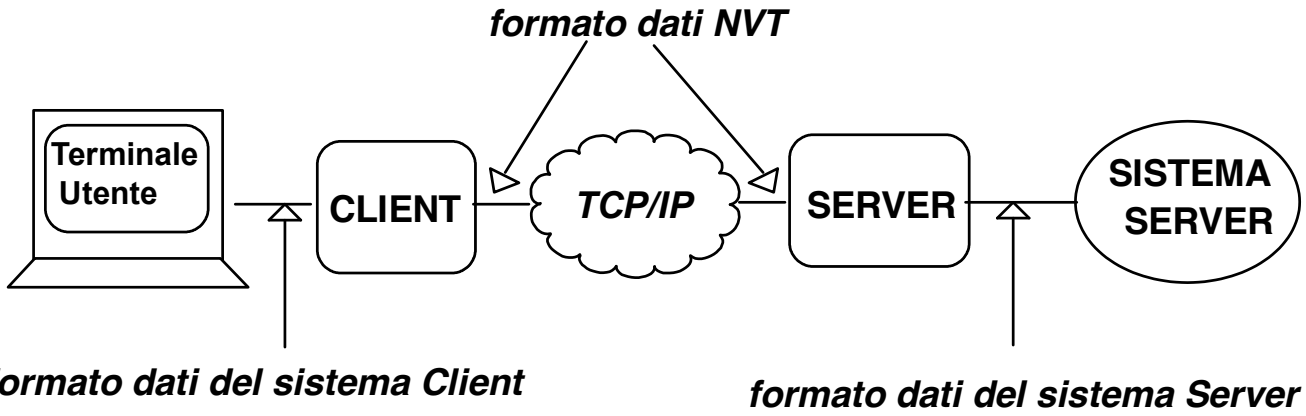
soluzione:
definizione di un **terminale virtuale**

La rete considera un unico tipo di terminale.
In corrispondenza di ogni stazione di lavoro, si effettuano la conversione da terminale locale a terminale virtuale e viceversa.
telnet, **rlogin** sono basati su questo modello
detto **NVT** (ossia **N**etwork **V**irtual **T**erminal)



Implementazione

Uso di formato **NVT (Network Virtual Terminal)**
All'inizio NVT prevede uso di rappresentazione a 7 bit USASCII
(*sequenze di comando* in byte con codifiche alte, 8^{o} bit settato,)



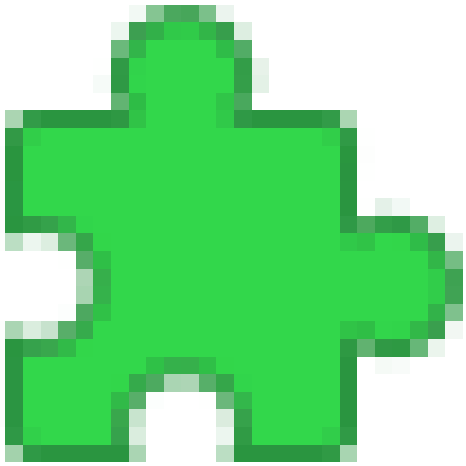
Client trasla caratteri utente nel formato NVT prima di inviarli al server

Server dal formato NVT nel formato remoto *viceversa al ritorno*

NEGOZIAZIONE
Possibilità di **negoziare** la connessione, sia alla *inizializzazione* sia *successivamente* per selezionare le opzioni del telnet
(comunicazione half- full- duplex, determinare il tipo di terminale, codifica 7-8 bit)

Protocollo per negoziare le opzioni è **simmetrico**, usa messaggi:
will X will you agree to let me use option **X**
do X I do agree to let you use option **X**
don't X I don't agree to let you use option **X**
won't X I won't start using option **X**
NVT definisce un tasto di interruzione concettuale che richiede la terminazione del programma

Caratteri di controllo NVT, USASCII



Funzioni di controllo NVT

(codificati con bit più significativo a 1)

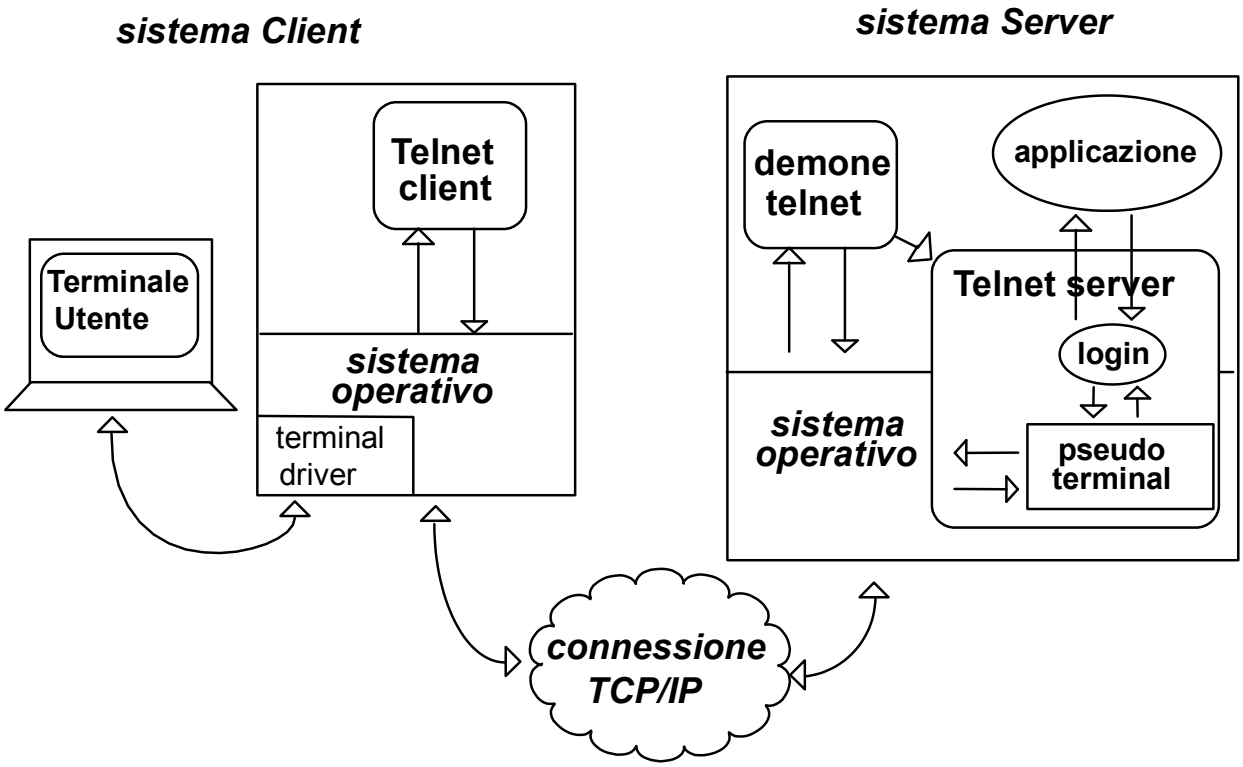
SEGNALE	SIGNIFICATO
IP	INTERRUZIONE DEL PROCESSO
AO	ABORT IN USCITA (SCARTA I CONTENUTI DEI BUFFER)
AYT	CI SEI? (TEST DELLA PRESENZA DEL SERVER)
EC	CANCELLA IL PRECEDENTE CARATTERE
EL	CANCELLA LA CORRENTE LINEA
SYNC	SINCRONIZZAZIONE
BRK	SOSPENSIONE TEMPORANEA (ATTESA DI UN SEGNALE)

invio di caratteri di controllo e funzioni di controllo insieme con i dati normali

Problema se lo stream dei dati è pieno

Uso di **dati urgenti** o fuori banda di TCP (out-of-band signal)

Implementazione



Pseudo-terminal può essere una
funzione del sistema operativo

Se il sistema operativo ha una *astrazione di pseudoterminale*
telnet ==> programma applicativo

vantaggio --> modifiche e controllo facili
svantaggio --> inefficienza

RLOGIN

Servizio di login remoto ⇒ login su un'altra macchina UNIX

```
rlogin lia02.deis.unibo.it
username:beppe
password:*******
```

Se l'utente ha una home directory in remoto
    accede a quel direttorio
Altrimenti,
    l'utente entra nella radice della macchina remota

Il servizio di rlogin UNIX supporta il concetto di trusted hosts.
Utilizzando i file
        .rhosts
        /etc/hosts.equiv
per garantire corrispondenze tra utenti.
    (uso senza password)

Problemi di **sicurezza** rlogin:
    - nell'uso di .rhosts ed hosts.equiv
    - password in chiaro

Caratteristiche RLOGIN

- utilizzo di una sola connessione TCP/IP
- conosce l'ambiente di partenza e quello di arrivo, ha nozione di stdin, stdout e stderr (collegati al client mediante TCP).
- esporta l'ambiente del client (es. il tipo di terminale) verso il server
- flow control: il client rlogin tratta **localmente** i caratteri di controllo del terminale (ctrl-S e ctrl-Q fermano e fan ripartire l'output del terminale) (in modo simile il ctrl-C)

out-of-band signaling per i comandi dal server al client (es. flush output per scartare dei dati, comandi per il resize della finestra e per la gestione del flow control)

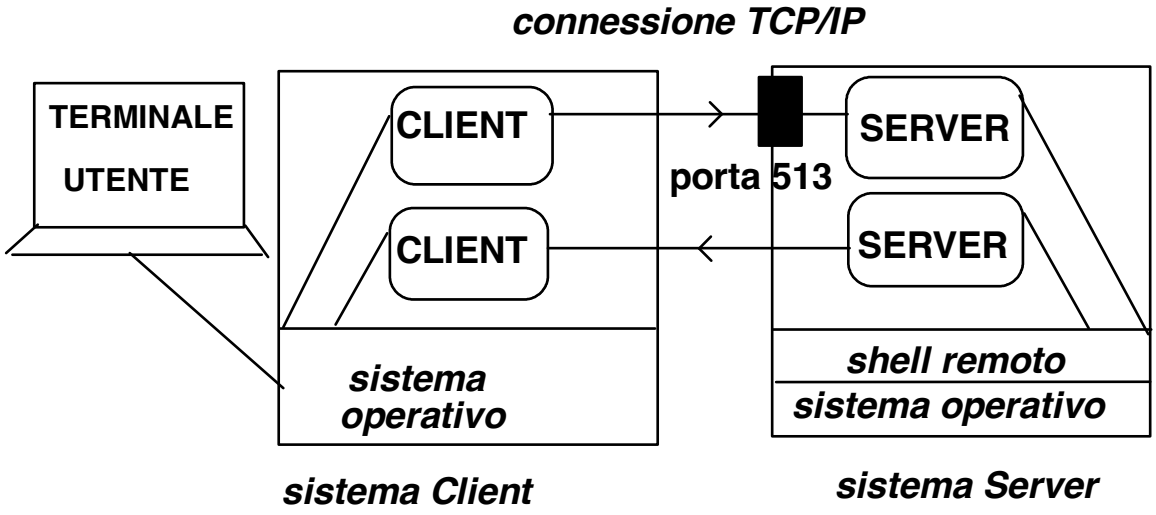
in-band signaling per i comandi dal cilent al server (spedizione dimensione finestra)

Implementazione

Client rlogin e Server remoto (server rlogind)

Il **client** crea una **connessione TCP** al server rlogind
Il **client** rlogin spezza le funzioni di ingresso/uscita
il genitore gestisce i caratteri che vanno allo shell remoto
il figlio gestisce i caratteri in arrivo dallo shell remoto

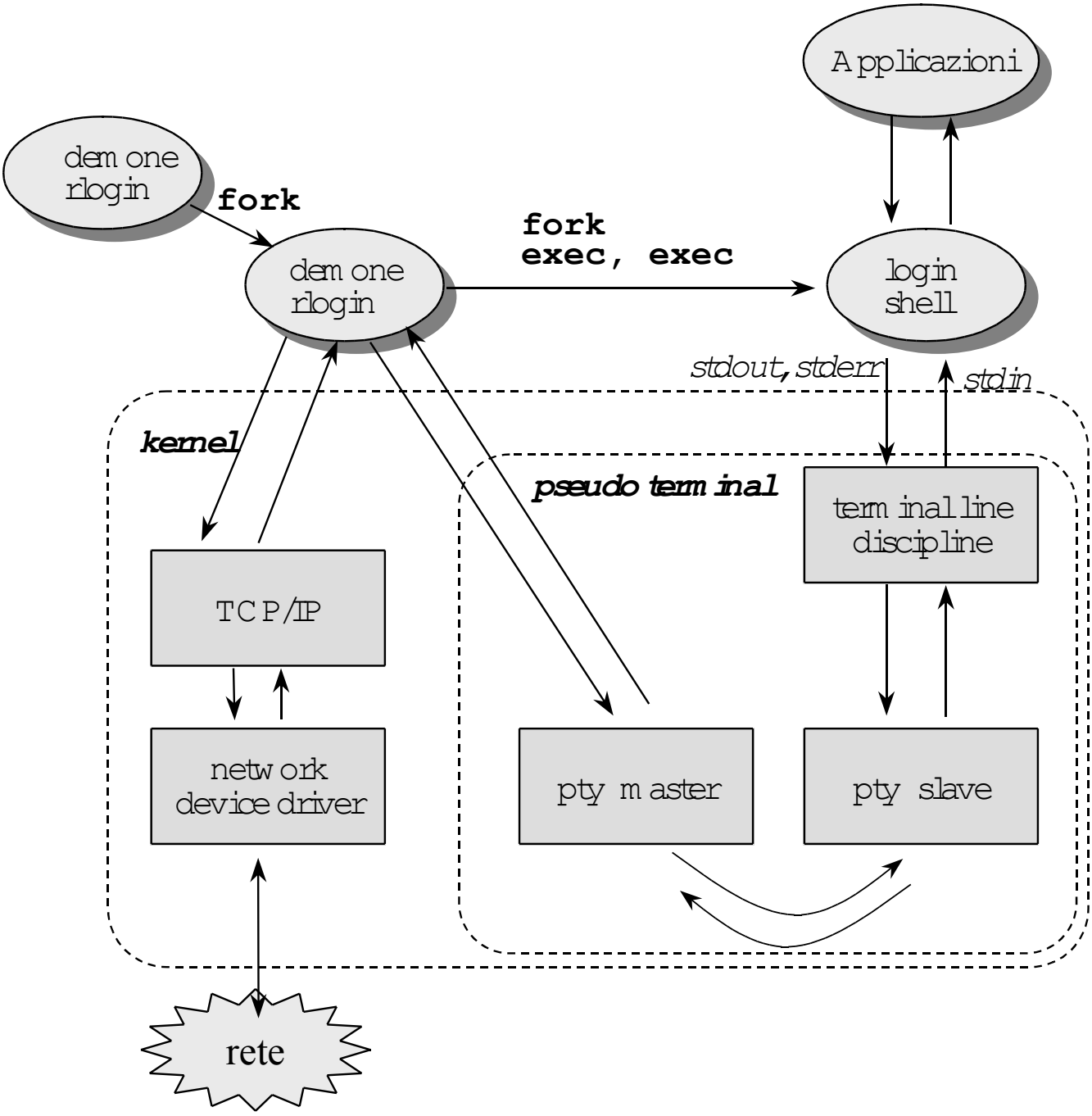
Il server si collega ad uno shell remoto
con coppia master-slave di uno pseudoterminale



Invocazione remota

Anche *rsh*
invoca l'interprete (shell) remoto UNIX
con gli argomenti della linea di comando
rsh nodo_remoto comando

Implementazione e pseudoterminal



EVOLUZIONI possibili

Modello di comunicazione

client / server avanzato
- parallelo
- stateful
- condivisione dello stato

Sicurezza
- autenticazione utenti tramite password
- liste d'accesso sugli oggetti
- domini di protezione tramite ruoli standard e relazioni mutue

Uso del server come framework di interazione

ACCESSO E TRASFERIMENTO FILE

Uso di TCP (affidabile ed orientato alla connessione)

ftp (file transfer protocol)

tftp (trivial file transfer protocol)

Permettono la copia di file nei due sensi.

Inoltre,

- Eseguito dai programmi applicativi o con accesso **interattivo** (si può richiedere la lista dei file di un direttorio remoto, o creare un direttorio remoto, etc.).

- Specifica del **formato** dei dati (rappresentazione): file di tipo testo o binario.

- Controllo **Identità** (login e password).

Comandi di trasferimento file

```
put local-file [remote-file]
```

memorizza un file locale sulla macchina remota

```
get remote-file [local-file]
```

trasferisce un file remoto sul disco locale

`mget` e **`mput`** utilizzano metacaratteri nei nomi dei file

altri comandi:

```
help, dir, ls, cd, lcd, ...
```

tfpt più semplice e con meno possibilità (uso di UDP)

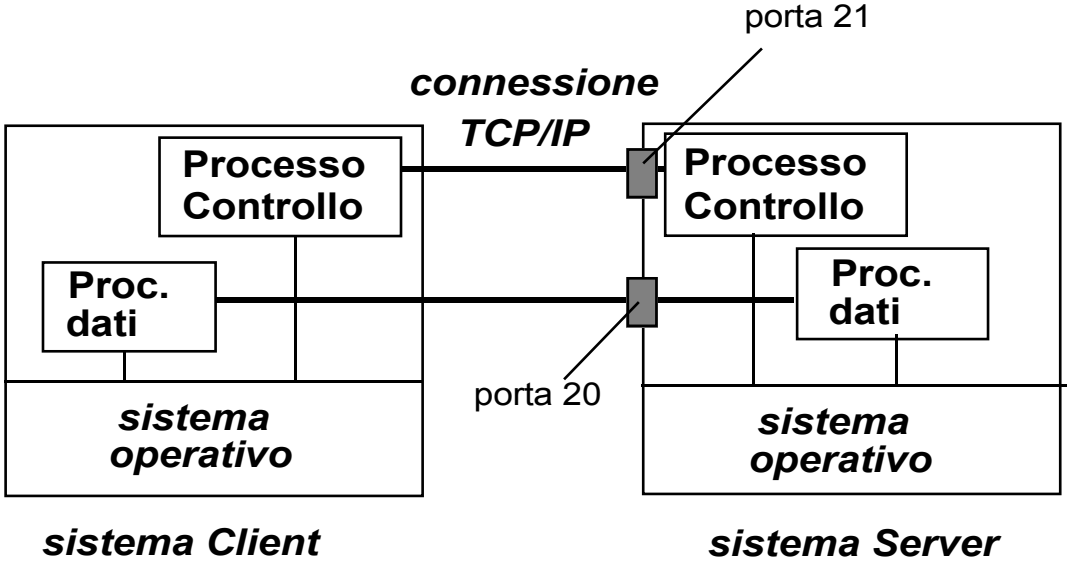
Esistono nodi server di ftp che sono contenitori di informazioni a cui si può accedere "liberamente"

Uso di ftp anonymous verso i server

Implementazione FTP

accesso concorrente da parte di più client ad un unico server ftp
uso di TCP per la connessione al server

Due collegamenti per ogni client e per ogni server:
UNA **CONNESSIONE DI CONTROLLO** e **UNA DI DATI**



Dettagli della implementazione:
Un processo **master** del server attende connessioni (processo **ftpd**, demone di ftp) e si crea uno **slave** per ciascuna connessione

Ogni slave è composto da:
- un processo che gestisce il collegamento di controllo con il client
  (persiste per tutta la durata del collegamento)
- un processo per il trasferimento dati
  (possono essere più di uno all'interno dello stesso collegamento)

Anche il client usa processi separati per la parte di controllo e di trasferimento dati

In certe macchine vi è un unico processo incaricato sia del controllo che del trasferimento dati (in generale sempre su diverse connessioni TCP)

Uso di numeri di porta tcp

TCP: entrambi gli estremi individuano una connessione

Nel caso di FTP due connessioni: una dati e una controllo:

Collegamento controllo
la porta di trasferimento lato server è fissa (21)
accordo sulla porta dalla parte del cliente (xxx)

Collegamento dati
la porta di trasferimento lato server è fissa (20)
porta da parte del cliente (yyy)

Client collegamento iniziale con server comunicando la
propria porta (xxx)
Se i servizi sono sequenziali,
la stessa porta xxx del cliente può essere usata per la connessione dati

I valori di porta passati al server rappresentano una forma di negoziazione senza la quale il servizio può non andare a buon fine

Quale formato per il passaggio delle informazioni di controllo? => uso di NVT

Confronto telnet ftp

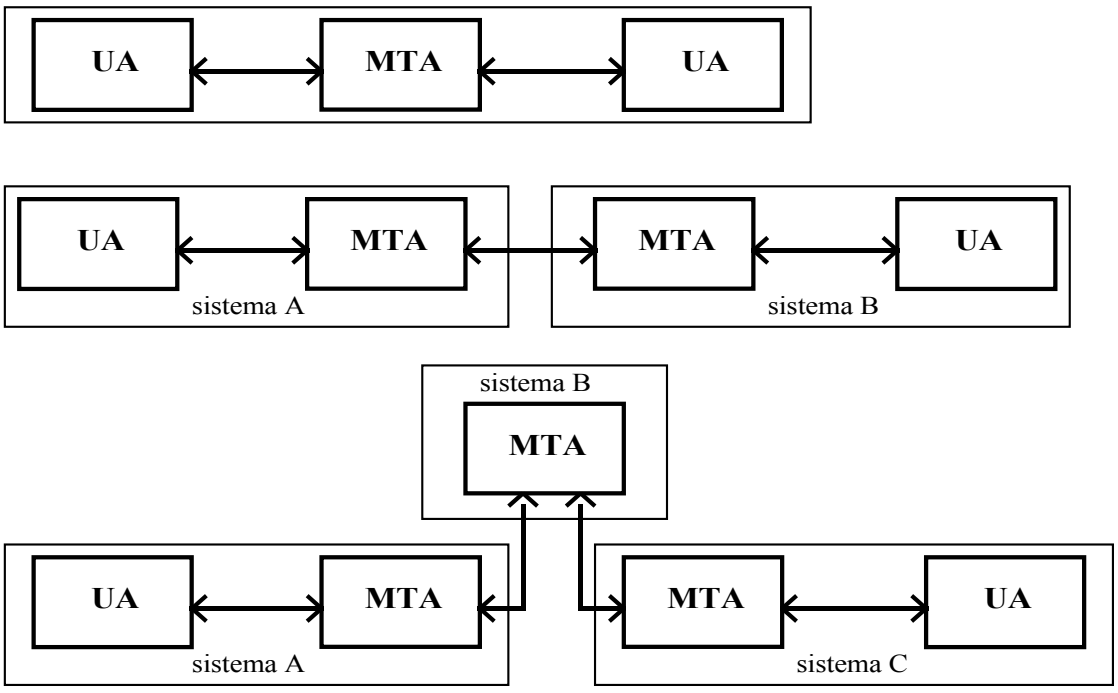
Servizio	FILE TRANSFER ftp tftp	VIRTUAL TERMINAL telnet rlogin
Oggetto	file	caratteri
Distribuzione	punto a punto	punto a punto
Protocollo	NVT	NVT

Architettura del servizio di mail

Uso di comunicazioni **punto a punto**
attraverso una rete di
User Agent (UA) e
Mail Transfer Agent (MTA)

Mail Transport Agent (MTA) trasferisce mail dal user agent (UA) sorgente a quello di destinazione

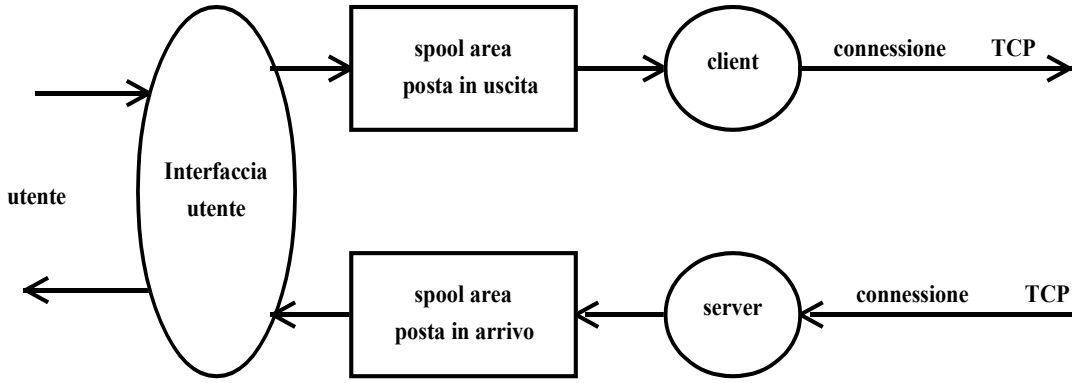
Diversi metodi di collegamento sistemi di posta elettronica



Uso di mailbox come area riservata ad un solo utente

posta elettronica come *sistema **asincrono***
- il mittente non aspetta il destinatario
- spooling

Componenti del servizio di posta elettronica



Un processo in background diventa il cliente

- mappa il nome della destinazione in indirizzo IP
- tenta la connessione TCP con il mail server destinazione
- Se OK, copia del messaggio alla mailbox remota

Protocollo SMTP (Simple Mail Transfer Protocol)

Un processo di trasferimento (in background)

controlla spool area

dopo un certo tempo se invio non OK

torna il messaggio al mittente

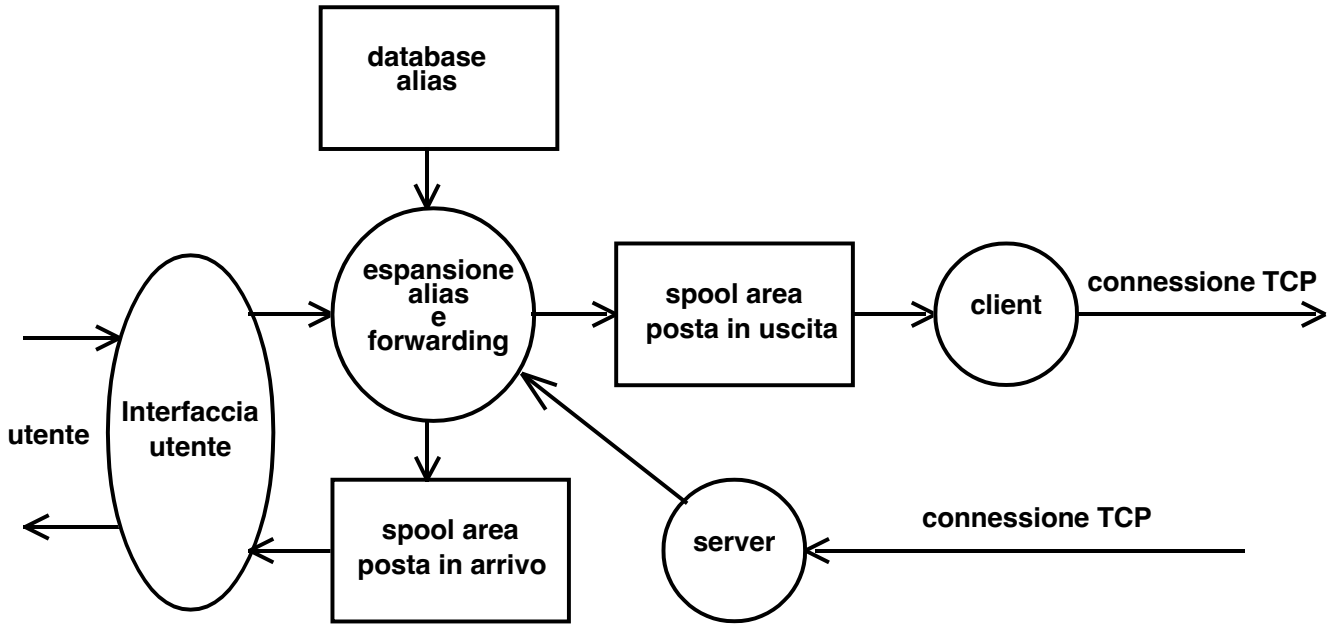
Indirizzi di mail

destinatario come

{ **identificatore IP nodo di destinazione**
mailbox sul nodo (nome login)
}

Altri indirizzi

mappaggio identificatori distinti in nomi di sistema
anche **pseudonimi (aliases)** e **mail forwarding**



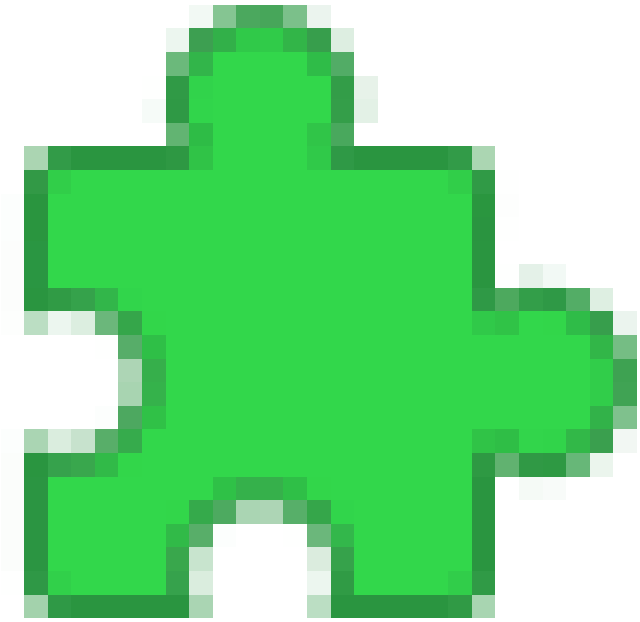
un utente può avere **più identificatori di mail**
ma anche
unico identificatore per un gruppo di destinatari

electronic mailing list anche con destinatari non locali
Esempio
nella mailing list di A, x mappato in y di B
nella mailing list di B, y mappato in x di A
Problema: *Ciclo senza fine*

Mail in INTERNET TCP/IP

Servizio di posta elettronica con

1) connessione di tipo **end-to-end diretto** (TCP-IP)
2) uso di mail gateway (macchine intermedie)



Evoluzione dello **standard** di **mail**
con **messaggi multimediali**
e **garanzie di sicurezza**

Formato dei messaggi

From:

```
To:
Date:
Subject:
Corpo:
```

`From:` indirizzo del mittente
`To:` mailbox cui il messaggio va recapitato
anche più indirizzi di destinazione
`Date:` la data di spedizione
`Subject:` il soggetto del messaggio

opzionali
`Cc:` copia destinatari
`Replay-To:` indirizzo per la risposta
`Message-Id:` Identificatore unico del messaggio

Corpo

il testo dei messaggi è in formato ASCII

Indirizzi di posta elettronica

varie possibili forme

```
From: acorradi@deis.unibo.it (Antonio Corradi)
```

postmaster mailbox del postmaster in ogni dominio
MAILER-DAEMON segnalazioni di problemi

Collegamento con il domain name

mailbox@subdomainN..subdomain1.top-level-domain

più sottodomini

```
acorradi@deis.unibo.it
acorradi@deis33.deis.unibo.it
```

Mail

Esempio di uso:

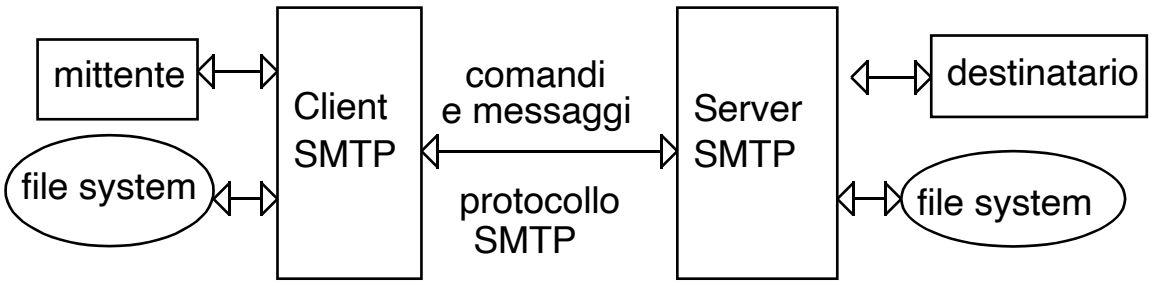
```
cesare deis32 ~ 19 >Mail beppe@deis.unibo.it
Subject:  Prova di mail

testo del mail

ctrl-D

beppe deis ~ 19 > Mail
Mail version SMI 4.1-OWV3 Mon Sep 23 07:17:24 PDT
1991  Type ? for help.
"/usr/spool/mail/beppe": 1 message 1 unread
>U  1 cesare              Sun May 21 16:48   14/296
Prova di mail

& return
Message  1:
From cesare Sun May 21 16:48:38 1995
Received: by deis.unibo.it (4.1/4.7); Sun, 21 May
95 16:48:37 +0200
From: cesare (Cesare Stefanelli)
Subject: Prova di mail
To: Beppe
Date: Sun, 21 May 95 16:48:37 MET DST
X-Mailer: ELM [version 2.3 PL11]
Status: RO

Testo del mail
```

Lettori di posta elettronica:
Mail, mail, elm, eudora, ...

protocollo SMTP

standard per il trasferimento della mail
Simple Mail Transfer Protocol RFC 821

Scambi di messaggi codificati tra un client ed un server



PROTOCOLLO **SMTP**
Comandi cliente ----------### Risposte server
Esempio

```
sender      'MAIL FROM:'   nome del mittente
receiver    'OK'

sender      'RCPT TO:' nome del destinatario
receiver    'OK'   abilitato

sender      'DATA'
               linee di testo del messaggio
sender      '< cr-lf> < cr-lf> '  fine messaggio
receiver    'OK'
```

I ruoli tra sender e receiver (o client e server) possono essere invertiti per trasmettere la posta diretta nel verso opposto.

COMANDI: parole composte di caratteri ascii:
RISPOSTE: composte di codice numerico di 3 cifre e testo

Codifica

La prima cifra codifica le interazioni
1xx Comando accettato
2xx Risposta positiva completa
3xx Risposta positiva intermedia
4xx Risposta negativa transitoria
il comando può essere ripetuto
5xx Risposta negativa permanente

La seconda cifra codifica le risposte
x0x Sintassi
x1x Informazione
x2x Connessione
x3x e x4x Codici non specificati
x5x Mail system (stato del receiver)

La terza cifra specifica più precisamente

Procedure di SMTP

Procedura di invio come *mail transaction*
MAIL TRANSACTION
fatta in modo da completare la trasmissione

Se tutto va bene OK
Se problemi
messaggi disordinati e ripetuti
(azioni di posta idempotenti ?)

S: MAIL FROM:<Smith@Alpha.ARPA>
R: 250 OK
S: RCPT TO:<Jones@Beta.ARPA>
R: 250 OK
S: RCPT TO:<Green@Beta.ARPA>
R: 550 No such user here
S: RCPT TO:<Brown@Beta.ARPA>
R: 250 OK
S: DATA
R: 354 Start mail input; end with <CRLF>.<CRLF>
S: Blah blah blah...
S: ...etc. etc. etc.
S: <CRLF>.<CRLF>
R: 250 OK

Return-Path: <@GHI.ARPA,@DEF.ARPA,@ABC.ARPA:JOE@ABC.ARPA>
Received: from GHI.ARPA by JKL.ARPA ; 27 Oct 81 15:27:39 PST
Received: from DEF.ARPA by GHI.ARPA ; 27 Oct 81 15:15:13 PST
Received: from ABC.ARPA by DEF.ARPA ; 27 Oct 81 15:01:59 PST
Date: 27 Oct 81 15:01:01 PST
From: JOE@ABC.ARPA
Subject: Improved Mailing System Installed
To: SAM@JKL.ARPA

This is to inform you that ...

MAIL FORWARDING
forward-path non corretto
251 User not local; will forward to ###forward-path###
551 User not local; please try ###forward-path###
VERIFYING AND EXPANDING
verificare di *user name* (VRFY)
espansione di mailing list (EXPN)

VRFY user-name

i) *250 'username completo' <indirizzo>*
ii) *251 User not local; will forward to <indirizzo>*
iii) *551 User not local; please try <indirizzo>*
iv) *550 That is a mailing list, not a user*
550 String does not match anything
v) *553 User ambiguous.*

EXPN <mailing-list>

S: EXPN Example-People
R: 250-Jon Postel <Postel@USC-ISIF.ARPA>
R: 250-Fred Fonebone <Fonebone@USC-ISIQ.ARPA>
R: 250-Sam Q. Smith <SQSmith@USC-ISIQ.ARPA>
R: 250-Quincy Smith <@USC-ISIF.ARPA:Q-Smith@ISI-VAXA.ARPA>
R: 250-<joe@foo-unix.ARPA>
R: 250 <xyz@bar-unix.ARPA>

OPENING E CLOSING

HELO <domain> <CR-LF>
QUIT <CR-LF>

RESET (RSET)

abort della transazione corrente; receiver deve inviare OK

TURN (TURN)

intenzione di scambio dei ruoli

USENET News

E' un insieme di **gruppi** di discussione.

Ogni gruppo riguarda un particolare argomento e permette di partecipare a una discussione su tale argomento, scambiando informazioni e facendo domande.

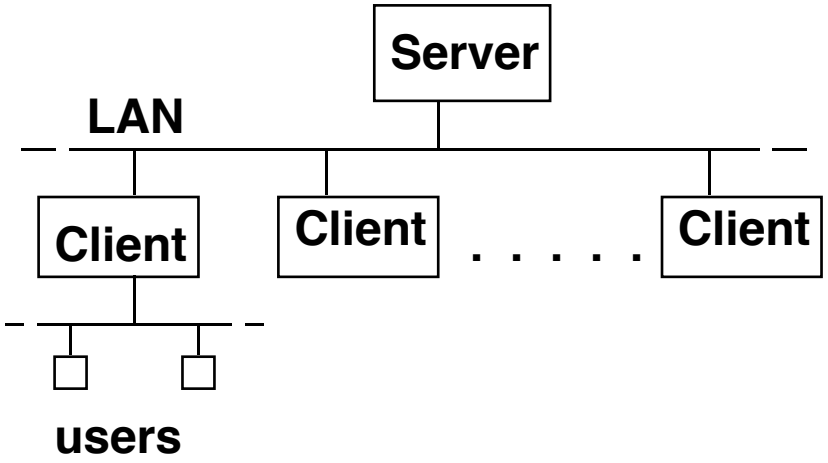
Insiemi aperti di interessi pubblici
GERARCHIE PRINCIPALI DI NEWS

comp	(COMPUTER)
misc	(MISCELLANEOUS)
news	(NEWS)
rec	(RECREATIVE)
soc	(SOCIETY)
sci	(SCIENCE)
talk	(TALK)
alt	(ALTERNATIVE)
bit	(BITNET)
biz	(BUSINESS)

SOTTOGERACHIE DI '*comp.unix*'
admin, aix, amiga, aux, internals, large, misc, programmer, question, etc ...

STORIA
1979 3 macchine uucp
1980 anews con due soli gruppi
1982 bnews

Architettura del servizio di NEWS



Nodo **client**:

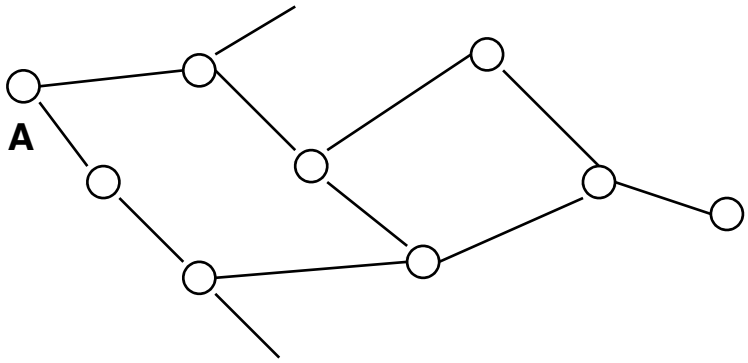
- un **client** di news mantiene le news
- presenza di **lettori** di news

Il **client** si coordina con il (i) **server** per ottenere le news

I client sono *strumenti per l'accesso applicativo alle news e consentono anche di inviare news ai gruppi di interesse.*

Uso di agenti con **TCP/IP** di connessione

News: uso di ***database coordinati*** per le informazioni



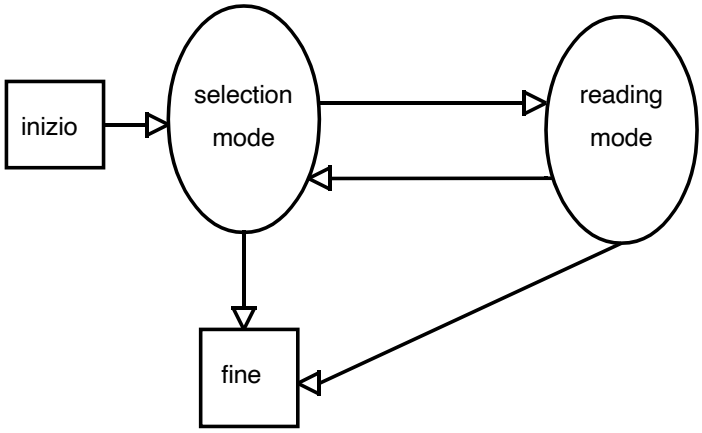
Implementazione

nn Lettore news - interfaccia utente
nntpd Demone protocollo TCP/IP

nn news reader

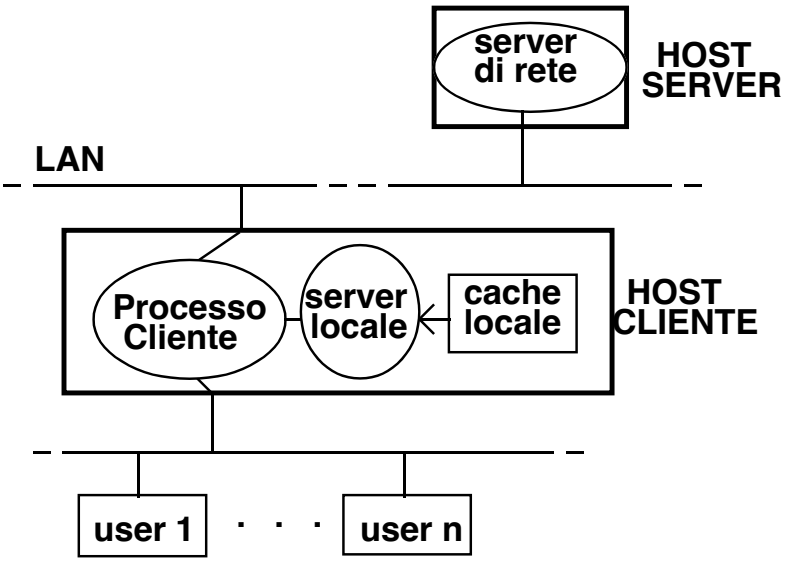
nn [opzioni] [news group | +folder | file]

Stati di nn:
Stato di **selezione** dei gruppi di news
Stato di **lettura** di un gruppo di news selezionato



Stati di funzionamento di nn

Possibile interconnessione



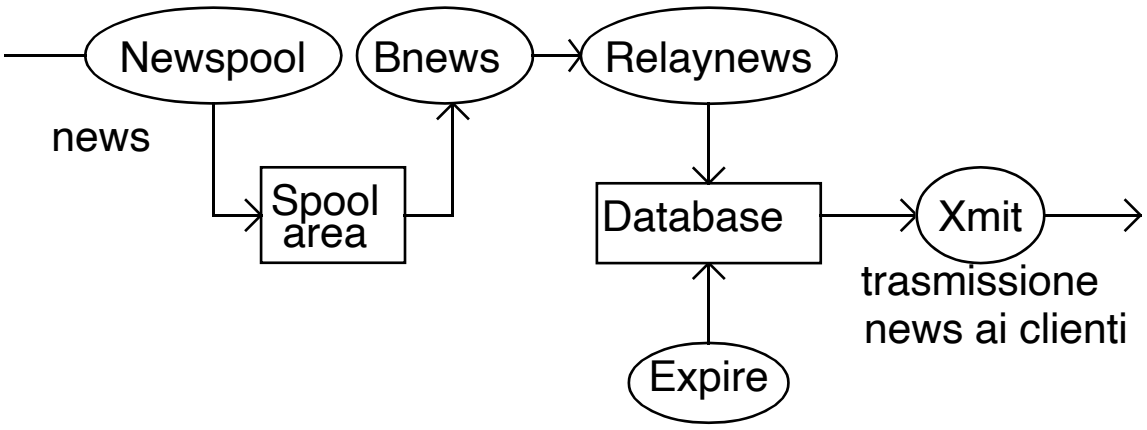
News SERVER

Funzioni del server:

- Accettazione delle news in ingresso
  (newspool, rnews)

- Realizzazione e gestione del database
  (relaynews, expire)

- Invio delle news richieste dai processi cliente
  (xmit)



Protocollo news:

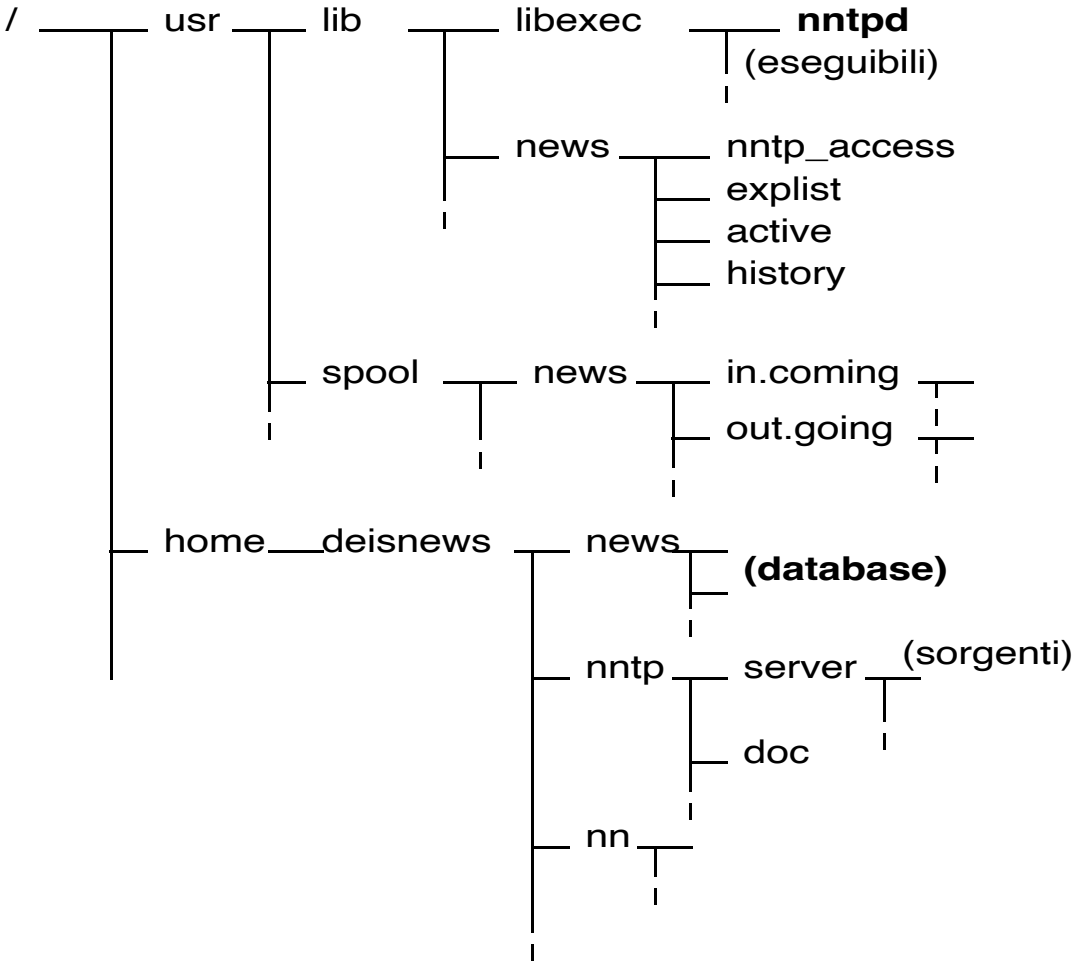
Il protocollo è *NNTP (USENET)*

Comandi cliente ----------### Risposte server

COMANDI: parole composte di caratteri ascii:
RISPOSTE: composte di codice numerico di 3 cifre e testo
In genere gli agenti si coordinano usando la **porta 119**

Esempio di file system

in cui si memorizzano le news



explist → eliminazione
history → pr evitare duplicazioni
active → gruppi ricevuti
in.coming → news da memorizzare
out.going → news da esportare

Implementazione

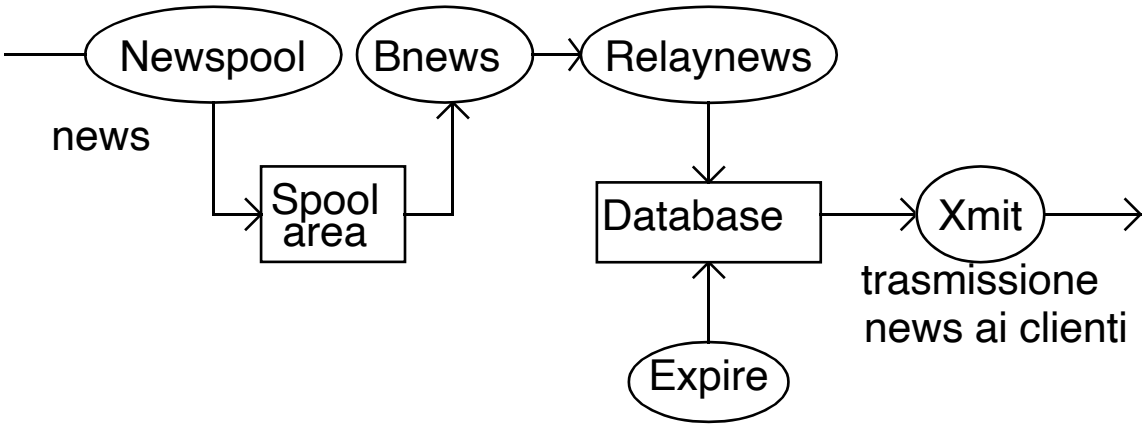
nn Lettore news - interfaccia utente
nntpd Demone protocollo TCP/IP

Funzioni del SERVER nntp
(**porta 119**)

Accettazione delle news in ingresso
(newspool, rnews)

Realizzazione e gestione del database
(relaynews, expire)

Invio delle news richieste dai processi cliente
(xmit)



Protocollo NNTP

Protocolli a negoziazione

Come smtp, così nntp (USENET)

comandi **e** risposte

il server restituisce una risposta al comando del client con il risultato dell'azione chiesta

comandi sono una parola di comando (ASCII)

più parametri separati e fine con carattere <CR> <LF>

risposte di testo e di stato

le risposte di testo

linee successive con un <CR> <LF>

le risposte di stato

stato dal server per l'ultimo comando

codice numerico di tre cifre

prima cifra successo o meno

1xx - messaggio informativo
2xx - comando ok
3xx - comando non ancora ok, richiesta del resto
4xx - comando corretto ma non eseguito
5xx - comando non implementato, o scorretto, o errore

seconda cifra categoria della risposta

x0x - messaggi di connessione, setup, e vari
x1x - selezione newsgroup
x2x - selezione articoli
x3x - funzioni di distribuzione
x4x - posting
x8x - estensioni non standard
x9x - debugging output

```
100 help text
190 through
199 debug output
200 server ready - posting allowed
201 server ready - posting not allowed
400 service discontinued
500 command not recognized
501 command sintax error
502 access restriction or permission denied
503 program fault - command not performed
```

PROTOCOLLO **NNTP**
Comandi cliente ----------### Risposte server

COMANDI: parole composte di caratteri ascii:
RISPOSTE: composte di codice numerico di 3 cifre e testo
In genere gli agenti si coordinano usando la port 119

CODICI NUMERICI DI RISPOSTA
Utilizzati per la gestione automatica delle risposte.

C: GROUP msgs
S: 211 103 402 504 msgs Your new group is msgs
C: ARTICLE 401
S: 423 No such article in this newsgroup
C: ARTICLE 402
S: 220 402 ###4105@xyz-vax.ARPA###
Article retrieved, text follows
S: (invio del testo da parte del server)
S: 205 XYZ-VAX news server closing connection. Goodbye

Confronto mail news

Servizio	POSTA ELETTRONICA	NEWS
Oggetto	messaggi	messaggi
Distribuzione	mailbox	database centralizzati
Protocollo	SMTP	NNTP

Network File System

Sistema Distribuito per **UNIX SUNOS**

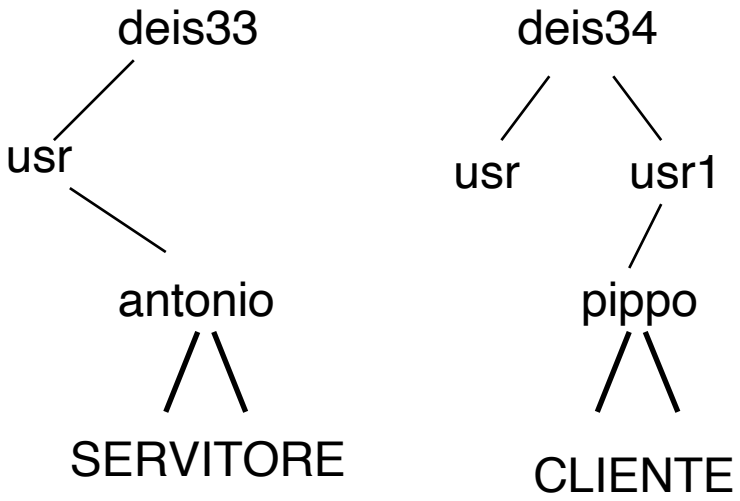
non TRASPARENZA della ALLOCAZIONE

il SUPERUTENTE monta
cioè decide di rendere visibile
una parte del file system di un nodo
su un altro file system

P.e. su deis34

```
mount deis33:/usr/antonio /usr1/pippo
```

A questo punto, tutta la gerarchia /usr/antonio è visibile dal file system come /usr1/pippo



Da deis34 si vedono i file di deis33
e scompare la visibilità del direttorio locale

TRASPARENZA nella COMUNICAZIONE

a livello utente ===>
l'UTENTE vede i file come se fossero LOCALI

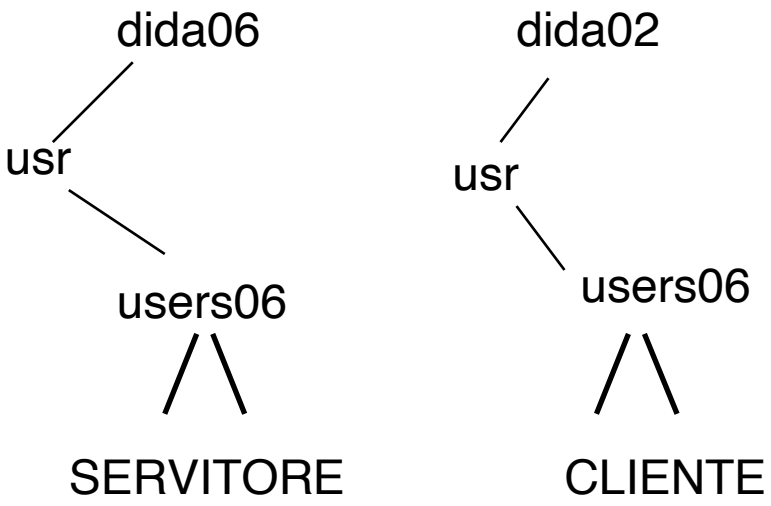
Ogni file system può contenere parti degli altri
I file relativi sono condivisi tra le diverse macchine

COSTO

Ogni cambiamento fatto su un file ==>
propagato dal file system servitore al cliente

In genere, si tende a mantenere la semantica di UNIX,
cioè propagazione di ogni cambiamento agli altri
utenti del file

Miglioramento attraverso l'uso di cache



Altri strumenti (BSD)

iniziali tutte con r

rcp copia dei file da remoto a locale e viceversa

ruptime verifica la presenza di nodi remoti
rwho uso di un demone rwhod che invia pacchetti broadcast (molto costoso)
rup presenza di nodi remoti

rexec esecuzione su un nodo remoto
uso di un demone **rexecd**

System Management a livello TCP

hostname identità del nodo corrente

netstat stato locale TCP
- **-a** comunicazione
- **-i** statistica
- **-r** tabelle di routing
- **-s**
- **-m** buffer
- **-I** monitoring

`ifconfig` esamina l'interfaccia con la rete

`ping` invia pacchetti ICMP al nodo remoto

`trpt` monitoring

`inetd` fa partire un server unico per i servizi locali

relativo ad RPC

`rpcinfo` informazioni sul port mapper

`nfsstat` informazioni su NFS

`spray` invio di RPC per un totale di 100K byte

strumento di monitoring di basso livello

`etherfind` estrae informazioni riguardo ai pacchetti sulla rete (invocabile solo da root)

`traceroute` invio di messaggi fornendo informazioni sul routing