# ADPnetwork: una rete per Amiga

di Andrea de Prisco

*Come anticipato lo scorso mese, a partire da questo numero presenteremo ADPnetwork, la nostra rete software per Amiga 500, 1000 e 2000 scritta interamente utilizzando l'ADPmttb 2.0, visto, anche quello, sul numero sorso.*

*Trattandosi di una realizzazione tutt'altro che banale, sia sotto il profilo teorico che implementativo, la presentazione del lavoro ci terrà impegnati per diverse puntate. Nella versione in cui vedrà la luce su queste pagine, tutti i collegamenti tra le macchine sfruttano l'interfaccia seriale dei singoli sistemi: in questa configurazione, ovviamente non andremo oltre gli aspetti puramente didattici della realizzazione a causa della bassa velocità di trasferimento che però potrà essere più che sufficiente per carichi leggeri. ADPnetwork è di fatto una piattaforma sulla quale costruire applicazioni distribuite, file system distribuiti, sistemi fault tollerant ed è indipendente, questo è molto importante, dalla natura del mezzo di comunicazione fisico adottato. Per essere ancora più precisi, il software di rete ADPnetwork è addirittura indipendente anche dalla macchina, l'Amiga, sulla quale attualmente gira: basta infatti scrivere l'mttb per un altro sistema multitasking (ad esempio Unix, OS/2), aggiungere o togliere un po' di include machine-dependent, ricompilare il tutto e il gioco è fatto.*

*Dopo la presentazione del vero e proprio software di rete (tutti i processi che costituiscono ADPnetwork), lasceremo la parola a Marco Ciuchini e Andrea Suatoni che hanno sviluppato un Handler e un NetServer (ovviamente «rigorosamente Amiga») per la rete, in modo da rendere visibile da ogni applicazione in esecuzione su qualsiasi macchina tutte le unità di ingresso uscita disponibili sull'intera rete. Ciliegina finale, sempre grazie all'handler, la possibilità di utilizzare la rete anche da WorkBench dove troveremo la nostra brava icona NET (vicino alle altre icone relative a floppy, harddisk, ram, rad) che una volta click-ata ci mostrerà le icone delle macchine attualmente in rete e click-ando su queste accederemo ai vari device disponibili sui nodi e, conseguentemente, ai file.*

*Tutto tramite mouse... non male, vero?*

## Un po' di storia

Lo schema di funzionamento «circolare» di ADPnetwork, come vedremo meglio in seguito, è tutt'altro che originale. Disponendo infatti su ogni macchina di una sola interfaccia di ingresso e una di uscita (nel nostro caso l'I/O seriale) tale forma di comunicazione è l'unica possibile volendo collegare più macchine tra di loro. Altre possibili tipologie (figura 1) sono quelle «a stella», in cui tutte le macchine sono collegate ad un unico nodo centrale, ad «interconnessione completa» in cui ogni macchina è direttamente collegata con tutte le altre macchine, ad «albero» dove i nodi sono collegati (e accessibili) gerarchicamente come i file di un file system, appunto, gerarchico. Non mancano tipologie più complesse, come il «mesh quadrato», «l'ipercube» o le strutture ad «Omega» o, ancor di più, tipologie miste in cui troviamo varie sottoreti con architetture diverse interconnesse a loro volta tra loro secondo le più svariate tipologie.

ADPnetwork, come detto, adotta uno schema di funzionamento «circolare» in cui ogni macchina ha un nodo precedente, dal quale riceve il flusso dei dati, e uno successivo al quale trasmette, o ritrasmette, dati. È chiaro a questo punto che ogni macchina analizzerà i dati ricevuti dalla rete o dovrà essere in grado di riconoscere messaggi per sé da inoltrare agli opportuni server, oppure da rimettere in circolo non riconoscendosi come destinatario. In questo modo è sia possibile che qualsiasi macchina dialoghi con qualsiasi altra macchina della rete, sia che in ogni istante (nel senso fisico e non solo in quello logico del termine) più macchine effettuino operazioni sulla rete. Quest'ultima caratteristica (non presente nemmeno in architetture ultra diffuse come Ethernet e Token Ring) è forse la più importante di tutta l'architettura di ADPnetwork, ed è implementata grazie al fatto che la struttura di comunicazione utilizzata è solo apparentemente condivisa da tutte le macchine. Infatti non succede che l'intero mezzo fisico appartiene a tutti i nodi (con la necessità quindi di arbitrare l'accesso in maniera centralizzata o distribuita, deterministicamente o non) ma ogni nodo è proprietario del collegamento alla macchina successiva, e quindi non deve dar conto alla rimanente rete delle sue operazioni. Un'occhiata allo schema di collegamento di figura 2 molto probabilmente vi chiarirà le idee più di mille parole.

L'attuale release funzionante di ADPnetwork, la 3.0, permette a qualsiasi processo in esecuzione su qualunque macchina di inviare messaggi a qualsiasi altro processo in esecuzione su qualsivoglia altra macchina collegata alla rete. Ogni messaggio può essere di lunghezza arbitraria e per inoltrarlo via rete il processo mittente dovrà naturalmente specificare il nodo destinatario, la porta mttb esistente su quel nodo (creata cioè dal processo destinatario) e consegnare il messaggio al software di rete. Sarà poi questo che, impacchettandolo opportunamente ed utilizzando l'interfaccia d'uscita verso la macchina «successiva» farà in modo (naturalmente con la complicità di tutti i processi di rete di tutte le macchine «attraversate») che il messaggio giunga a destinazione e sulla giusta porta.

Attualmente ADPnetwork prevede anche alcune forme di tolleranza ai guasti, ivi comprese cadute del supporto fisico o più semplicemente spurie di trasmissione che potrebbero cambiare il contenuto di un msg. Per caduta del supporto fisico intendiamo anche (a dire

il vero «soprattutto») accidentali sconnessioni delle macchine nel bel mezzo di una operazione sulla rete. L'abbiamo visto coi nostri occhi (e vi assicuriamo che la cosà è affascinante) provocando artificialmente una sconnessione mentre una macchina stava stampando su video (a seguito di un TYPE NET: ecc. ecc.) un file remoto. Non essendo possibile la comunicazione (i connettori erano stati brutalmente staccati) la visualizzazione si è arrestata per riprendere esattamente dallo stesso punto, e senza battere ciglio, pochi attimi dopo aver ripristinato il collegamento. Come vedremo più approfonditamente in seguito, tale meccanismo è implementato attraverso due appositi processi mttb di ADPnetwork (Timer e Replay) che ritrasmettono automaticamente le porzioni di messaggio che non risultano essere arrivate a destinazione.

### *Schema circolare*

Come detto prima, ADPnetwork è indipendente dal mezzo fisico adottato per mettere in comunicazione più macchine. L'unica cosa necessaria, è il collegamento «circolare» in cui l'interfaccia out della macchina «j» è collegata all'interfaccia di ingresso della macchina «(j+1) mod n» dove «n» è il numero delle macchine in rete. Nel caso minimale di due macchine in rete, basterà un collegamento dalla macchina A alla macchina B e, viceversa, uno da B ad A. Se le macchine sono tre avremo un cavo da A a B, uno da B a C e uno da C ad A. E cosi via per collegamenti di più macchine. Un'altra caratteristica particolare di ADPnetwork è che non è necessario informare i singoli nodi dei nomi degli altri nodi esistenti in quanto ogni accesso a macchine remote viene di volta in volta «tentato» confidando sull'effettiva esistenza dello stesso. Ciò significa, ad esempio, che tanto l'ingresso in rete, quando l'uscita dalla stessa, non è subordinato ad operazioni di servizio atte ad informare le altre macchine del cambiamento. L'unica cosa, banalmente, necessaria, è il fatto che ogni macchina per fare parte della rete deve essere:
(1) fisicamente collegata alla struttura;

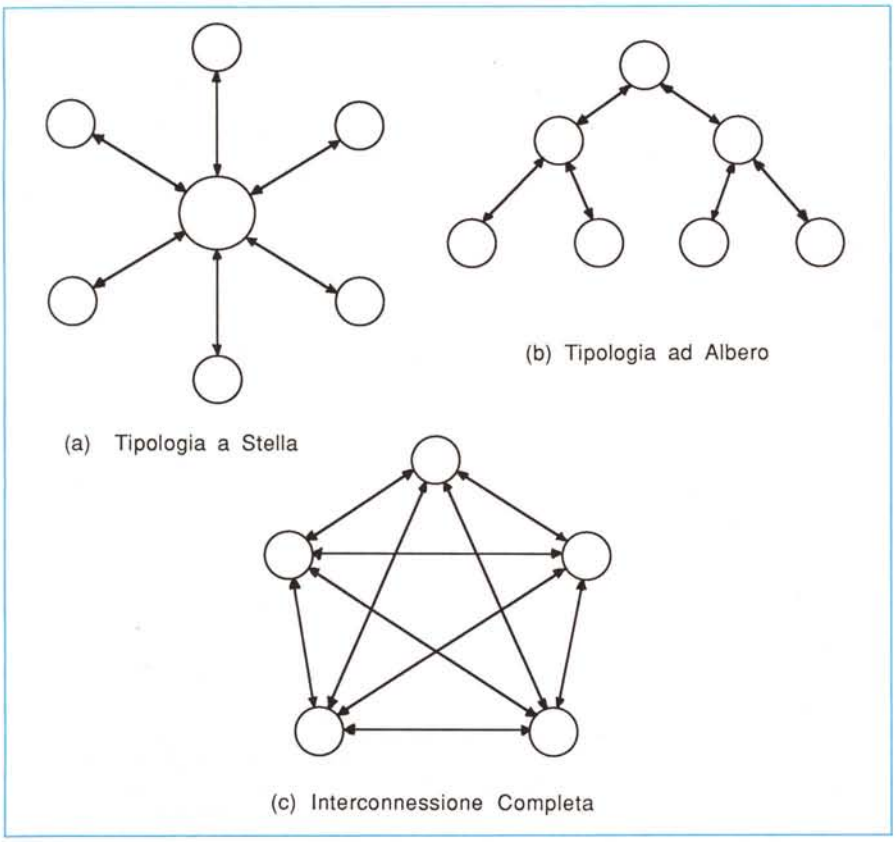


*Figura 1 - Alcune tipologie di collegamenti in rete.*

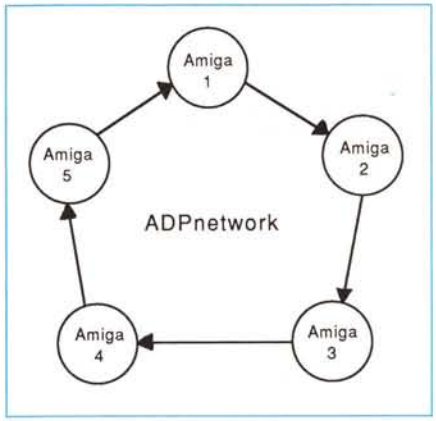


*Figura 2 - Esempio di collegamento «circolare» o, più propriamente, a «maglia».*

(2) abbia in esecuzione i processi di gestione per la rete;
(3) abbia un nome unico formato da caratteri alfabetici.

Il nome della macchina è passato ai processi del software di rete in esecuzione sul nodo e serve sia per ricevere messaggi che per trasmetterli. Infatti è necessario che qualsiasi messaggio in viaggio sulla rete sia sempre accompagnato da un nome del mittente, un nome del destinatario più altre informazioni di servizio che avremo modo di analizzare meglio in seguito.

Poniamoci ora dal punto di vista di un singolo nodo e cerchiamo di spiegare a parole e nel modo più comprensibile possibile (si spera) il funzionamento a grandi linee di ADPnetwork. Diciamo, per semplicità, che il nodo in questione sia «Platone» e che sulla rete esistano altri due nodi instanziati come «Socrate» e «Pitagora». Il collegamento circolare è mostrato in figura 3.

Sull'interfaccia di ingresso di Platone arriva ad un certo punto un messaggio di Socrate per Pitagora. Platone non lo riconosce come proprio e non fa altro che reinviare il messaggio senza alcuna modifica sul suo canale d'uscita verso la macchina successiva (che in questo caso è Pitagora, ma poteva pure non esserlo). In questo primo esempio, i processi di Platone hanno permesso la comunicazione tra due nodi non direttamente connessi. Da notare, anche se sicuramente non ce n'è bisogno, che i processi di gestione alla rete girano in background su tutte le macchine e quindi un eventuale operatore sulla macchina Platone che sta ad esempio scrivendo una lettera o divertendosi col DeLuxe Paint non si accorge minimamente

del fatto che in quel momento la sua macchina sta «servendo» la rete. Questo almeno in teoria, infatti, utilizzando la seriale come interfaccia prêt-à- porter su tutte le macchine, un lieve rallentamento delle operazioni si avvertirà comunque. Ma come detto prima la scelta della seriale è solo di comodo e nulla vieta di realizzare delle schede elettroniche aggiuntive in modo da demandare il riconoscimento dei messaggi e relativa eco sulla rete ad hardware apposito ed esterno all'architettura Amiga.

Tornando allo schema-esempio di figura 3, grazie all'intercessione di Platone, Pitagora riceve il messaggio inviato da Socrate. Diciamo che a questo punto, Pitagora decide di rispondere al messaggio preparando una risposta, appunto, per Socrate. Quello che fa è

# ADPnetwork story

L'idea di connettere in rete più Amiga mi balenò nella mente praticamente appena saputo che la nuova macchina Commodore aveva un sistema operativo multitasking. Parlo dunque di qualcosa come almeno tre anni e mezzo fa o anche più. Ovviamente poi tra il dire e il fare c'è di mezzo il mare, ma anche la famiglia, il lavoro, gli altri impegni e progetti. Questo ritardò non solo la progettazione della rete vera e propria, ma anche un propedeutico studio della macchina (o, meglio, del multitask della macchina) che cominciò non prima di un anno fa. Nacque così l'ADPmttb col quale fare in multitask anche le cose più difficili diventa un vero e proprio gioco da ragazzi.

Per la realizzazione del software di rete, grazie all'ADPmttb, me la potevo certamente cavare da solo, ma per scrutare nei meandri più oscuri di Amiga, per collegare ad esempio il mio software ai programmi già esistenti mi serviva la mano di un vero e proprio gruppo di lavoro, soprattutto per evitare di mandare avanti il progetto per alcuni anni. Era il 19 marzo di quest'anno, meno di 7 mesi fa, e decisi di utilizzare MC-Link per raccogliere consensi, ma soprattutto collaboratori. Il mio primo messaggio in area Amiga, per motivi di «modestia» conteneva anche un piccolo bluff: il gruppo di lavoro non esisteva ancora...

Mostriamo qui di seguito alcuni stralci di messaggi tuttora presenti nell'area Amiga di MC-Link che pubblichiamo per tutti i nostri lettori ancora non telematicamente preparati. Altri contatti li ricevei per posta elettronica, tra cui quelli con Ciuchini e Suatoni e devo dire che senza MC-Link lavorare in perfetta sintonia e a distanza sarebbe stato praticamente impossibile (per non parlare della dispendiosità). Buona lettura «storica».

---

*Amiga*
*Msg# 2187, 19/03/89 08:31 [1263]*
*Da: MC0258 Andrea de Prisco*

---

*Oggetto:* **ADPnetwork For All Amigas**

Io e il mio gruppo di lavoro stiamo cominciando a studiare una rete software di Amiga 1000, 2000, 500, in modo da rendere disponibile, da qualsiasi postazione tutti i device di tutte le postazioni. Ovviamente in maniera completamente trasparente, così tutti i programmi in esecuzione sulle macchine, senza alcuna modifica potranno leggere, scrivere o stampare su qualsiasi altra macchina. Con l'aggiunta poi di un hardware piccolo piccolo, possiamo fare in modo che spegnendo una qualsiasi macchina (anche accidentalmente) la rete continui a funzionare sulle rimanenti macchine, a condizione che non si richiedano accessi sulla macchina spenta. Alternativamente l'operatore prima di spegnere dovrà sconnettere i suoi due connettori di ingresso/uscita sulla rete e collegarli tra di loro. Se vi sembra interessante per partecipare o solo per saperne di più fatemi un fischio (mi raccomando qui in area!). Ciao...

*adp*

PS: 'ADPmttb ha già trovato lavoro...

---

*Amiga*
*Msg# 2188, 19/03/89 18:02 [115]*
*Da: MC2176 Maurizio Maglio*

---

*Oggetto:* **Capito un tubo...**

Che vor di'??? Non ho capito un tubo Andrea me lo rispieghi? Come rete software? Quali macchine? Boh...

Starfy!

---

*Amiga*
*Msg# 2189, 19/03/89 19:46 [201]*
*Da: MC2837 Andrea Borghi*

---

*Oggetto:* **Re: ADPnetwork For All Amiga**

Per ADP:

In riferimento all'invito a partecipare alla costruzione di una rete Amiga vorrei rendere nota la mia diponibilità a partecipare al lavoro e attendo ulteriori notizie.

*Andrea*

---

*Amiga*
*Msg# 2190, 19/03/89 19:59 [2704]*
*Da: MC58492 Franco Saiu*

---

*Oggetto:* **Finalmente Si Parla di Janus**

(...)

Per Andrea, devo confessare che anche io non ho capito esattamente il progetto, comunque ogni cosa che riguarda Amiga è interessantissima, quindi non tenerci all'oscuro dei risultati, per la collaborazione potrò esprimermi dopo aver messo maggiormente a fuoco il progetto.

*Ciao a tutti*
*Mercurio*

---

*Amiga*
*Msg# 2191, 19/03/89 20:25 [314]*
*Da: MC4474 Alex Bianchini*

---

*Oggetto:* **Re: Netsoft**

Io invece credo di aver capito tutto... OTTIMO! Mi pare un progetto molto interessante (una specie di LAN per Amiga). Come suggerimenti, vedrei bene un kit di sviluppo soft per la rete, e la possibilità di mettere un modem tra due macchine, in modo da non dover per forza disporre di più Amiga in un locale.

*Alex*

---

*Amiga*
*Msg# 2192, 19/03/89 20:31 [170]*
*Da: MC2250 Rodolfo Rossi*

---

*Oggetto:* **Re: ADPnetwork For All Amiga**

Per ADP:

Mi aggrego alle persone che hanno già dato la loro disponibilità a partecipare al lavoro per la costruzione di una rete Amiga.

*Rodolfo Rossi*

---

*Amiga*
*Msg# 2195, 19/03/89 22:32 [1320]*
*Da: MC0258 Andrea De Prisco*

---

*Oggetto:* **ADPnetwork**

Grazie per l'eco vistosa alla mia idea di rete per Amiga. Non è possibile interporre due modem tra due Amiga in rete a meno che non si intenda collegare solo due macchine. Comunque 1200/2400 baud mi sembrano un po' pochi. La rete avrà una architettura circolare: ogni Amiga è collegato solo con due macchine, la precedente e la seguente ma, ripeto, ogni macchina può accedere a qualsiasi altra macchina semplicemente indirizzando opportunamente il device richiesto. Ogni macchina che riceve un ordine dalla macchina precedente se è diretto a lei lo esegue altrimenti passa lo stesso comando alla macchina successiva. E così via. Se una

scaricare tale messaggio sull'interfaccia di uscita che questa volta è direttamente connessa all'interfaccia d'ingresso di Socrate, dunque giunge immediatamente a destinazione. È importante sottolineare che in tutti i casi, ogni macchina oltre a non sapere a priori i nomi delle macchine collegate, tantomeno conosce l'ordine della disposizione. Quindi l'attuale situazione di Pitagora che invia a Socrate è esattamente la stessa della precedente in cui Socrate spediva a Pitagora; l'unica cosa che una macchina mittente può fare è inoltrare il messaggio sulla sua interfaccia d'uscita sperando che la macchina destinataria esista.

Tutti i lettori a questo punto si chiederanno cosa succede realmente quando una macchina invia un messaggio ad una macchina inesistente, vuoi per un

macchina che spedisce un comando vede ritornare il suo stesso comando vuol dire che nessuna macchina ha il device richiesto: in questo caso l'operazione abortisce, ma né la rete, né le altre macchine battono ciglio.

*adp*

---

*Amiga*
*Msg# 2709, 13/05/89 18:52 [580]*
*Da: MC0258 Andrea De Prisco*

---

*Oggetto:* **ADPnetwork V. 1.0 (Evviva!)**

L'ADP SOFTWARE è lieta, anzi lietissima, di annunciare che ha preso vita il primo embrione di sistema operativo distribuito di rete ADPnetwork. Attualmente stanno funzionando in rete tre Amiga ma l'espansione del numero di macchine è completamente trasparente al sistema di rete distribuito. Maggiori dettagli sull'avvenimento (ma soprattutto sugli sviluppi) sui prossimi numeri di MC microcomputer o, se sono interessati anche i componenti di questo forum, anche direttamente (e più speditamente) in quest'area.

*adp*

---

*Amiga*
*Msg# 2829, 24/05/89 08:09 [1316]*
*Da: MC0258 Andrea De Prisco*

---

*Oggetto:* **A proposito di ADPnetwork**

Avevo detto che «scucivo» un po' di informazioni in anteprima e non l'ho ancora fatto. Bene, ADPnetwork 1.0 funziona (quasi) perfettamente ma non è ancora implementato come device dos compatibile di Amiga. Ovvero non esiste ancora un vero e proprio «net:» né un net.handler (anche se sta arrivando!!!) ma solo il software di rete scritto interamente in ADPmttb (multi tasking tool box). Attualmente ho implementato, per provare la rete, due comandi. Il primo, NET-COPY, permette il trasferimento di un file residente in un qualsiasi device di qualsiasi macchina in un qualsiasi device di qualsiasi macchina. Così dalla macchina 1 posso ordinare di copiare un file che sta nella ram disk della macchina 3 sulla stampante della macchina 18 o sull'HD della macchina 6. Funziona! Il secondo comando, semplicemente NET, permette invece di eseguire un qualsiasi comando su qualsiasi macchina e ricevere i risultati sulla macchina dalla quale si è impartito il comando NET. Così posso chiedere la directory di qualsiasi device, ma anche cancellare rinominare e... formattare qualsiasi cosa da qualsiasi macchina. È un comando un po' pericoloso, quindi da usare con le opportune cautele, ma serve solo al sottoscritto per giocare col suo nuovo giocattolo...

*adp*

---

*Amiga*
*Msg# 2907, 01/06/89 07:00 [1013]*
*Da: MC0258 Andrea De Prisco*

---

*Oggetto:* **ADPnetwork News**

Chiedo scusa a tutti gli utenti Amiga per le mie numerose e continue assenze (in scrittura) in quest'area. Come saprete sto lavorando (sodo) alla rete Amiga che, grazie alla preziosa collaborazione del gruppo di lavoro (attualmente: Marco Ciuchini, Andrea Gozzi, Andrea Suatoni e... adp), conto di presentare sulle pagine di MC già dal numero di settembre (che va in lavorazione a luglio, quindi non c'è molto tempo). Il software di rete è già «sufficientemente» funzionante (posso eseguire da qualsiasi macchina qualsiasi comando di quella macchina e/o trasferire file tra due qualsiasi device di due qualsiasi macchine impartendo l'ordine anche da una macchina intermedia) mentre aspetto con ansia il nuovo processo di gestione della seriale (più veloce e scattante...) e il net.handler da inserire nella mountlist per rendere il tutto utilizzabile da qualsiasi programma (in grado di vedere nuovi device, praticamente tutti). A risentirci...

*adp*

---

*Amiga*
*Msg# 2911, 01/06/89 22:15 [692]*
*Da: MC2250 Rodolfo Rossi*

---

Oggetto: **Re: ADPnetwork News**

C'è una cosa che mi sfugge del progetto ADPnetwork, ovvero la sua praticità a parte l'uso per la stesura di articoli. Sì, perché se da una parte è un tentativo di costruire una rete per Amiga, non è l'unico nella sua specie, e non credo che potrà essere utilizzato commercialmente, dal momento che sul mercato è già presente Ethernet per Amiga da circa un anno, ed oltretutto ad un prezzo abbordabilissimo, ed Ethernet è uno standard mondiale, non una customizzazione. Inoltre Ethernet permette uno scambio di dati ad una velocità tale che in nessuno modo una rete via seriale Amiga potrebbe raggiungere.

È comunque da elogiare il suo valore propedeutico.

*Doctor J.*

---

*Amiga*
*Msg# 2913, 02/06/89 07:29 [514]*
*Da: MC0258 Andrea De Prisco*

---

*Oggetto:* **ADPnetwork ed Ethernet**

No, no, hai proprio capito male. ADPnetwork ha SOLO scopo didattico: far vedere come sia possibile pensare anche a cose serie su una macchina che tutti spacciano per videogioco evoluto. E poi non fare finta di non sapere le cose: ne abbiamo parlato in redazione faccia a faccia e sai anche che la seriale serve solo come interfacciamento provvisorio per testare la rete (l'importante è che funziona, non quanto corre!). Poi vedremo come espanderci...

*adp*

---

*Amiga*
*Msg# 3533, 08/09/89 10:56 [1858]*
*Da: MC0258 Andrea De Prisco*

---

*Oggetto:* **C1 - Text e ADPnetwork**

A proposito di ADPnetwork, colgo l'occasione per anticiparvi alcune nuove feature, tra cui l'utilizzo via workbench (appare l'icona NET) della rete e la tolleranza ai guasti ormai perfettamente funzionante. Pensate che se durante una operazione sulla rete (esempio il type di un file presente sull'HD di un'altra macchina) si sconnettono brutalmente i fili di collegamento tra due o più macchine, lo scroll del file semplicemente si arresta momentaneamente per riprendere esattamente dallo stesso punto non appena qualcuno (smette di rompere e) rimette ogni connettore al suo posto. Per quanto riguarda l'utilizzo via WB, una volta bi-clickata l'icona NET si apre una finestra contenente un'icona per ogni macchina collegata in rete. Clickando su una di queste troveremo una icona per ogni device di quella macchina (ad esempio DF0, RAM, RAD, DH0, DF1, ecc. ecc.) e clickando ancora su uno di questi device vedremo comparire le icone presenti in quella unità. A questo punto sempre col mouse possiamo tanto lanciare applicazioni residenti su macchine remote oppure spostare icone da una macchina ad un'altra ed ogni altra operazione normalmente possibile da WB (tranne ovviamente rinominare qualche macchina o qualche device). Che ve ne pare?

*adp*

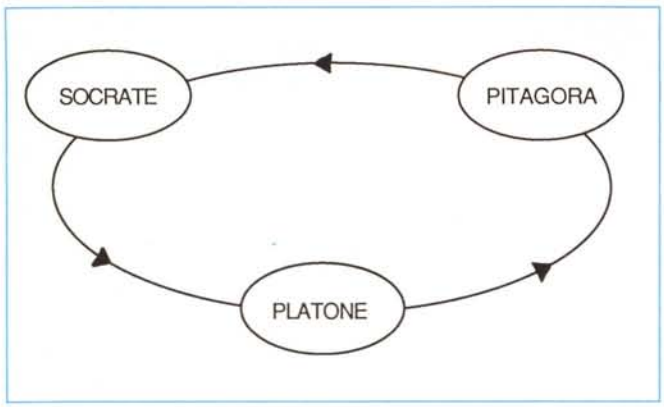


*Figura 3*
*Tre macchine in rete.*

errore di digitazione («Pitagola» in luogo di «Pitagora») vuoi perché il destinatario finora vivo e vegeto ha deciso, per cause a noi ignote, GURU compresi!, di uscire dalla rete. La risposta è quantomai banale: infatti come detto prima ogni messaggio circolante sulla rete è sempre accompagnato da un nome del mittente e da un nome del destinatario, quindi un messaggio diretto ad un destinatario inesistente sarà semplicemente scartato da tutti i nodi e giungerà nuovamente al mittente. Quindi ogni macchina, prima di re-inoltrare sulla rete un messaggio, oltre a controllare che il destinatario sia diverso dal nome proprio, deve controllare anche che il mittente non sia se stesso: se ciò, di contro, si verifica, vorrà dire che il messaggio ha fatto tutto il giro dell'architettura circolare con conseguente deduzione dell'inesistenza del destinatario in oggetto.

### *Struttura dei pacchetti*

Introduciamo a questo punto i cosiddetti pacchetti di rete, veri e propri veicoli di informazione trasmessa, che d'ora in poi chiameremo frame per non confonderli con i (già esistenti) pacchetti DOS o DOS packet che dir si voglia.

Infatti la spedizione di un messaggio da una macchina ad un'altra non avviene «sparando» direttamente l'oggetto della trasmissione sul canale d'uscita, ma avviene tramite una codifica in frame che grazie appunto alla frammentazione permette di minimizzare i fallimenti di trasferimento. Se spedissimo interi file sulla rete così come sono, in caso d'errore di trasmissione occorrerebbe rispedire nuovamente l'intero file, mentre frammentando la spedizione, basterà re-inoltrare solo i frame giunti non correttamente a destinazione. Naturalmente la frammentazione implica anche un aumento di informazione trasmessa, resa necessaria al destinatario per ricomporre il messaggio originario man mano che arrivano i frame.

In figura 4 è mostrato lo schema di un frame di rete. Cominciamo col dire che questi hanno lunghezza variabile tra 128 e 5120 byte. Ciò significa che messaggi lunghi meno di 5 K viaggeranno su un solo frame (di lunghezza appunto variabile) mentre per messaggi più lunghi si opererà la frammentazione. In questo caso se «n» sono i frame necessari per la spedizione, i primi «n-1» saranno lunghi 5 K, mentre l'ultimo avrà nuovamente lunghezza variabile a seconda di quanti byte rimangono da spedire. Facciamo un esempio: bisogna spedire un messaggio lungo 13 K da una macchina ad un'altra. Sono necessari tre frame, i primi due da 5 K l'uno, il terzo da 3 K. Semplice, no?

Vediamo ora in dettaglio quali informazioni sono contenute in ogni singolo frame di rete. Il primo campo, di un byte, contiene appunto la lunghezza del frame espresso in multipli di 128 byte. In questo campo ci sarà dunque un numero compreso tra 1 e 40 (40×128=5120). Seguono 10 byte per il nome del destinatario del frame e 10 per il mittente, necessari, come detto, sia per recapitare i frame che per «dedurre» che il destinatario non esiste. Le rimanenti informazioni sono di servizio ovvero sono necessarie al software di rete per controllare la trasmissione e la ricomposizione del messaggio originario. Il primo di questi campi indica il tipo di messaggio: come vedremo meglio in seguito messaggi di tipo diverso potranno circolare sulla nostra rete e saranno trattati in maniera diversa. Segue un byte per un primo check sum sui primi 22 byte del frame: questo campo è utilizzato per risincronizzarsi su un frame valido qualora si «perdesse per strada» qualche byte trasmesso. Dopo il byte di check sum troviamo un altro byte, detto chop byte, che indica quanti byte in coda al messaggio sono non significativi: questo si manifesta tutte le volte (ovvero quasi sempre) che il messaggio trasmesso non riempie perfettamente l'ultimo frame inviato. Il campo

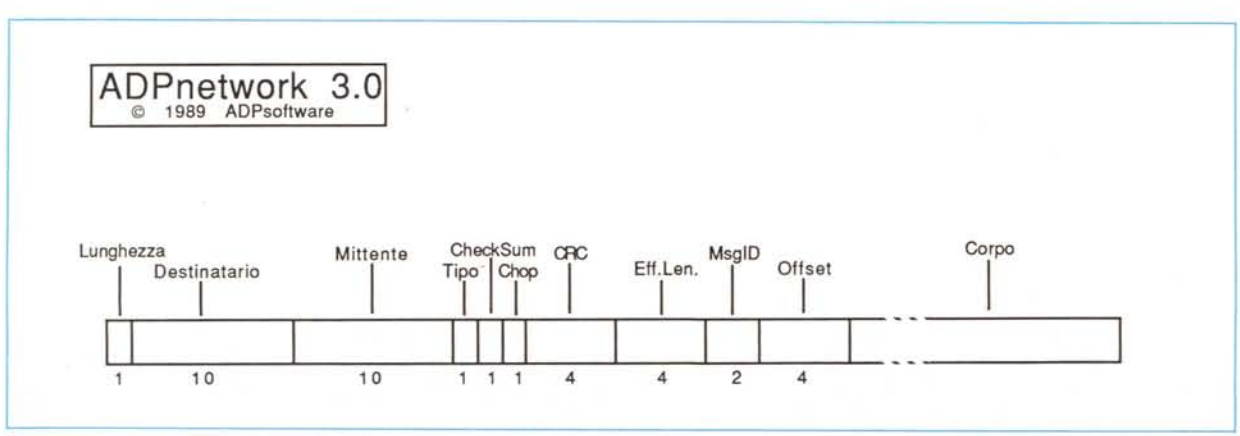


*Figura 4 - Formato Frame di Rete.*

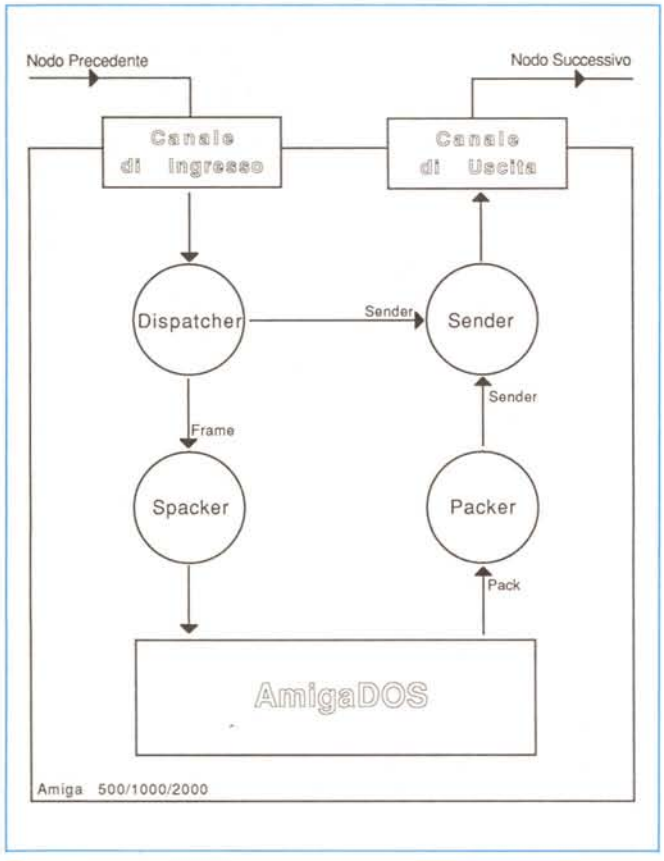


*Figura 5*
*I quattro processi di base di ADPnetwork.*

CRC (Cyclical Redundance Code) controlla invece possibili errori di trasmissione sulla vera e propria porzione di messaggio trasmessa. Gli ultimi tre campi servono per ricostruire il messaggio originario man mano che arrivano i frame. I primi due (costanti su ogni frame di uno stesso messaggio) indicano l'effettiva lunghezza del messaggio originario prima della frammentazione e l'identificatore di messaggio che viene incrementato, localmente ad ogni macchina, ogni nuova richiesta di spedizione su rete. L'ultimo campo, Offset, indica a partire da quale byte il corpo di questo frame va posizionato: nessuna assunzione è fatta circa la sequenzialità dei frame di un messaggio. Infatti frame arrivati male verranno rispediti solo dopo la prima spedizione dell'ultimo frame e quindi è sempre necessario sapere ciò che arriva a quale porzione del messaggio originario corrisponde.

Per finire, tutti i rimanenti byte del frame contengono la parte di messaggio spedita.

## *Il Software di Rete*

Per Software di Rete (SDR) intendiamo i processi lanciati in background su ogni macchina atti a permettere la comunicazione tra due qualsiasi nodi. Utilizzando l'SDR è possibile a due processi qualunque lanciati su due macchine distinte di scambiare agevolmente messaggi. È su questo SDR che Marco Ciuchini e Andrea Suatoni hanno installato i loro Net-Handler e NetServer per realizzare una sorta di file system distribuito permettendo tanto l'utilizzo sotto WB della rete quanto l'accesso a file remoti da shell e da ogni programma commerciale purché questo sia stato realizzato all'origine secondo canoni di programmazione «puliti». Tanto per non fare nomi, il nostro fiore all'occhiello italiano, C1-Text della Cloanto Italia, caricato su una macchina collegata in rete, nei requester «Aprire documento» o «Memorizzare documento» mostra un nuovo button «NET» tramite il quale possiamo leggere e salvare file su altre macchine.

L'SDR è attualmente formato da diversi processi cooperanti (scritti tutti, ovviamente, in ADPmttb 2.0) e per il momento vi mostreremo il funzionamento dei quattro modulil più importanti: Packer, Sender, Dispatcher e Spacker (beh, in effetti un nome più corretto per questo processo sarebbe UnPacker...). Il loro schema di cooperazione è mostrato in figura 5.

Poniamoci allora dal punto di vista di una macchina che deve spedire un messaggio ad un processo in esecuzione su un'altra macchina e vediamo cosa succede. L'ipotesi di collegamento è sempre quella di figura 3, che poi è quella che ci ha accompagnato in tutti questi mesi di sperimentazione e realizzazione qui in redazione. Allora, diciamo che Socrate vuole spedire un messaggio lungo 23 K a Pitagora. Ciò che deve fare è mandare un messaggio al processo Packer in esecuzione sulla sua stessa macchina, contenente il nome del destinatario («Pitagora») il tipo del messaggio (per il momento ignoreremo questo campo) la lunghezza del messaggio, 23552 pari cioè a 23×1024 byte, e il messaggio vero e proprio cioè i 23552 byte da spedire. Ricevuto il messaggio, il Packer provvede a spezzare i 23552 byte in 5 porzioni, quattro delle quali lunghe circa 5 K e l'ultima circa 3 K. Il circa è dovuto al fatto che ogni frame è lungo un multiplo di 128 byte, compreso però l'header che contiene le informazioni trattate nel precedente paragrafo. Ogni frame costruito dal Packer è passato al processo Sender che provvede alla effettiva spedizione sul canale di uscita. Lo stesso Sender riceve sulla medesima porta anche frame provenienti dal Dispatcher che, ricevendoli dal canale di ingresso, ma non riconoscendoli come destinati a quel nodo, li inoltra alla macchina successiva. Questa sarà l'operazione svolta da Platone che, come mostrato in figura 3, si trova tra Socrate e Pitagora. All'interno di quest'ultima, man mano che giungono i frame al suo Dispatcher, riconosciuti come propri, i frame sono inviati al processo Spacker che provvede a «riincollare» le 5 porzioni di messaggio arrivate in sequenza. Non appena il processo Spacker si accorge di aver ricomposto interamente il messaggio originario (e se ne accorge sapendo da ogni frame la lunghezza totale e la posizione relativa dei vari pezzi arrivati) provvede a spedirlo al processo opportuno, funzione del «tipo messaggio» che non abbiamo ancora trattato. Comunque per il momento è tutto. Mentre qui in redazione continuiamo a fare gli ultimi ritocchi «ottimizzatori» al SDR il sottoscritto e all'Handler di rete l'accoppiata vincente Ciuchini-Suatoni, vi diamo appuntamento al prossimo numero per un più approfondito commento al funzionamento dei processi e ai meccanismi utilizzati per implementare la tolleranza ai guasti.

MC