# NTG System Broker: Sommario

## Indice argomenti

Per una piccola introduzione di NTSYSBRK guardate dopo questo indice.





## Introduzione

### Obiettivi e possibili utilizzi del del programma

Per sommi capi, le possibilità offerte da NTSYSBRK sono:

- Eseguire in automatico, applicazioni DOS o Windows, sulla stessa macchina, o su altre macchine, in multitasking, oppure in modalità sequenziale.
- Poter eseguire applicazioni DOS o Windows da ambienti operativi differenti. Ad esempio da UNIX o OS/2, senza che sia installato il protocollo TCP/IP e i servizi RSH / REXEC.
- Inviare comandi DDE, o impostare variabli DDE di programmi che supportano il DDE in modalità Server (Excel, Winword, Program Manager, ecc...). Questo si può fare sia sulla stessa macchina, che inviare questi comandi ad applicazioni in esecuzione su altre macchine di una LAN.

- Inviare sequenze di tasti ad applicazioni in esecuzione sulla stessa macchina o su altre macchine della LAN, che siano applicazioni DOS o Windows.

**L'utilizzo più interessante è in una LAN, in ambienti multitasking o in attività di computing distribuito, dove è possibile effettuare delle manutenzioni sulle macchine client/server remote, eseguendo altre applicazioni tramite NTSYSBRK, da ambienti operativi differenti.**

## A chi è rivolto

Il programma è rivolto a:

- System Manager o Amministratori di Rete, perché possano meglio controllare le macchine della rete da loro gestita, eseguire elaborazioni su varie macchine, effettuare operazioni di manutenzioni su tutte le macchine.
- Programmatori che vogliano eseguire applicazioni DOS o Windows dai loro programmi. Infatti NTSYSBRK è programmabile dall'esterno attraverso stringhe di comandi in linguaggio NTF, di facile costruzione.

## Modalità di esecuzione

NTSYSBRK, è una applicazione che può essere utilizzata in queste modalità:

- COMMAND LINE: Passando all'applicazione i parametri. Da Windows 3.1 si può fare tramite "File Esegui.....".
- FILE ESTERNO di comandi: Se si passa al programma un solo parametro, il cui nome sia un file con estensione .NTF, allora i comandi verranno presi dal file in questione. Il limite del file è di 16Kb, senza limite nella dimensione della singola linea.
- RESIDENTE: Può essere lasciato in esecuzione (come i vecchi TSR del DOS per fare un esempio, ma è una cosa ben diversa), in attesa che gli vengano mandati dei comandi attraverso i canali di input supportati.

 Canali di input supportati

## Ambienti operativi supportati

- WINDOWS 3.11
- WINDOWS NT 3.5
- WINDOWS '95

### Requisiti di funzionamento

- Richiede la presenza di VBRUN300.DLL
- Risorse: 300Kb di RAM (alla partenza) fino a 500kb
- CPU: E' sufficiente un 386sx a 20mhz, il minimo con cui funziona windows 3.1

 Prossime versioni di NTSYSBRK e storia del prodotto

# Struttura del manuale

## Manualistica in formato hypertestuale

Il manuale d'uso dei software NTG viene fornita prevalentemente in formato hypertestuale WINDOWS HELP.
Quando lo si ritiene necessario, viene fornito anche in formato HTML (eventualmente come modulo aggiuntivo).

## Struttura multilinguistica

Dal momento che i prodotti NTG sono realizzati per un mercato internazionale, la lingua standard utilizzata è l'inglese. Per favorire la traduzione dei manuali, l'identificativo dei topics di ogni argomento è in inglese, il contenuto del topic può essere anche in un'altra lingua. Ciò permette anche il collegamento da un manuale ad un altro manuale NTG, risparmiando spazio.
Ecco perché, se utilizzerete le funzioni di ricerca troverete insieme termini in varie lingue (inglese ed eventualmente un'altra lingua) che spesso significano la stessa cosa.
Quando ci è possibile allegheremo anche in manuali nella nostra lingua naturale, cioè l'italiano.

## *Avvio NTSYSBRK*

Si rimanda all'esecuzione in un oggetto in architettura NCO.

***Approfondimenti:***

- Avvio di NCO
- Per terminare l'esecuzione di NTSYSBRK

# Avvio di NCO

## Avvio senza parametri

Se non viene passato nessun parametro, come in tutte le applicazioni in architettura NCO, appare un piccolo Help del programma, con le informazioni sulla versione del prodotto. Dalla maschera è possibile consultare il manuale completo ONLINE, premendo l'icona "?" o il tasto F1.

## Avvio con parametri in command line

Questa è la modalità standard di funzionamento di NTSYSBRK. Vengono passati i parametri, corrispondenti a transazioni NTF, una di seguito all'altra divisi da spazi. Per limiti di windows la linea di comando non deve essere più lunga di 127 caratteri. Se c'è questa possibilita, si consiglia di utilizzare la possibilità di passare come parametro il nome di un file di comandi.

```
Esempio:
ntsysbrk.exe @C=EXEC(FILENAME=WINWORD.EXE)
```

## Avvio con un file di comandi come parametro

Passare come parametro il nome del file, senza spazi. Il file sarà un file di testo contentente transazioni in linguaggio NTF.

```
Esempio:
ntsysbrk.exe comandi.ntf
```

## Avvio da File Manager

(Si veda il paragrafo dedicato al File Manager).

## Comportamento del programma all'avvio con parametri

Una volta letti i parametri ed eseguite le transazioni in essi contenuti, il programma termina, salvo sia stato il comando @C=@STAY. In questo caso NTSYSBRK rimane residente, fino a quando non riceve, attraverso i canali di input supportati, o tramite chiusura del processo, o tramite TASK MANAGER, il comando @C=@QUIT.

***Approfondimenti:***

- Canali di input
- Errori di un NCO
- Architettura NCO
- Linguaggio NTF
- Comando @STAY
- Avvio da File Manager

# Canali di INPUT

In questa versione i canali di input supportati sono:

## Input via COMMAND LINE

E' il canale che per primo viene letto. Le modalita' sono 2:

- Specificare uno o piu' comandi NTF, come parametri del programma ntsysbrk.exe, in command line. [divisi da uno spazio]. In ambiente Windows 95 e Windows NT il programma NTSYSBRK.EXE può essere eseguito da una shell. Da Windows 3.1 si può fare tramite la funzione "File Esegui..." del Program Manager. La riga non può superare i 127 caratteri di lunghezza.

```
Esempio: ntsysbrk @C=EXEC(FILENAME=NOTEPAD.EXE)
```

- Dichiarare il nome del file da cui prendere le istruzioni.

```
Esempio: ntsysbrk c:\comandi.ntf
```

## Input via FILE ESTERNO di comandi

Creando un file esterno di nome NTSYSBRK.NTF nella directory corrente. Questo canale viene utilizzato se si è "comandato" precedentemente all'applicazione di rimanere residente. Periodicamente NTSYSBRK verifica l'esistenza del file e ne legge il contenuto (cancellandolo una volta letto). A questa forma di input viene associato il broker NETF.
La differenza tra questo broker e quello che segue, consiste nel fatto che può essere cambiata dall'utente la directory dove verificare se esiste il file con estensione NTF (e nome dell'oggetto). Può anche essere una directory di rete. Dal momento che la verifica dell'esistenza del file viene fatta ogni 4-5 secondi, ciò non crea un carico eccessivo sulla rete. Il limite del file è di 16Kb, senza limite nella dimensione della singola linea.

### Attivazione del BROKER NETF

**Attenzione !! E' un broker opzionale (non abilitato quando si installa NTSYSBRK), con una modifica nelle configurazioni, con questo comando da inviare a NTSYSBRK:**

```
@C=REGPUT(OBJECT=BRKNETF VALUE=1)
```

## Input via FILE di comandi nella directory TEMP

Come sopra, ma NTSYSBRK verifica la presenza del file NTSYSBRK.NTF in una directory stabilita non cambiabile. Quella dichiarata nella variabile d'ambiente TEMP usata da Windows sul computer in cui viene eseguito il programma. Questo e' il canale di default di input.
A questa forma di input viene associato il BROKER FILE.

## Creazione di una directory TEMP condivisa in rete

Questa tecnica è quella più facilmente utilizzabile per far comunicare tra loro, in una LAN, macchine che utilizzano sistemi operativi differenti (DOS, UNIX, MS/DOS), ma che condividono il File System. Creando una directory TEMP (o con un altro nome) condivisa

sul computer dove è in esecuzione NTSYSBRK (in ambiente windows NT, windows '95 o windows for workgroup 3.11) , ed impostando questa directory come la TEMP per il computer, è possibile inviare messaggi al programma NTSYSBRK in esecuzione sulla macchina creando, dagli altri computers in rete, un file di nome NTSYSBRK.NTF che conterrà delle transazioni per NTSYSBRK.
Periodicamente NTSYSBRK (se il broker FILE è attivato) verifica se questo file è stato creato. Se esiste, lo legge, lo cancella ed esegue i comandi in esso contenuti.
L'operazione di condivisione della directory viene effettuata da File Manager, se il computer può condividere directory in rete.

### Caso di computer solo client. Utilizzo del broker NETF

Se il computer è solo un client, la tecnica sopra riportata, può essere utilizzata con il broker NETF ed impostando come directory da verificare periodicamente una directory del computer server, modificando la variabile interna @NP.

```
Esempio: @NP="\\\\TIGER\\TEMP"
```

Variabile @NP
Comando ROUTE

### Attivazione del BROKER FILE

**Attenzione !! E' un broker opzionale (ma abilitato in automatico all'installazione di NTSYSBRK), con una modifica nelle configurazioni, con questo comando da inviare a NTSYSBRK:**

```
@C=REGPUT(OBJECT=BRKFILE VALUE=1)
```

## Input via DDE

E' possibile inviare dei comandi a NTSYSBRK via DDE. Il nome del Dde server è lo stesso del programma e il Topic è "System". Non è necessario specificare un nome di Item. La sintassi con cui inviare i comandi è la stessa che per gli altri canali di input.
A questa forma di input viene associato il BROKER DDE.

## Input via MESSAGGIO interno di Windows

E' possibile mandare un insieme di comandi in linguaggio NTF, divisi da spazi o dal carattere 13 tramite le API dei messaggi, cioè SendMessage e PostMessage, con questi parametri:

- Codice messaggio: 1125, cioè WM_USER + 101.
- wParam: 0 oppure un handle ad un blocco di memoria globale allocato con GlobalAlloc e del tipo GMEM_SHARE + GMEM_ZEROINIT, contenente un insieme di comandi in linguaggio NTF. Alla fine della stringa può esserci (opzionalmente) il catattere 0 che non viene considerato.
- lParam: Se 1, allora NTSYSBRK si deve occupare di cancellare il blocco di memoria passato tramite wParam.

### Tabella dei broker

Riepiloghiamo la lista dei broker disponibili nell'architettura NCO, con le loro ID di riferimento.

- API: Api interne, l'oggetto è contenuto nell'applicazione.
- FILE: Comunicazione via FILE condiviso, directory TEMP.
- WMSG: Windows Message Api. Attualmente solo in lettura. Non è possibile spedire messaggi con questo broker. E' possibile farlo con l'utility esterna WSENDMSG.EXE

Utility NWSENDM.EXE

- NETF: Comunicazione via FILE condiviso, in directory specificata dall'utente, anche di rete.
- DDE: IPC di Windows.

## Utilizzo di protocolli di trasporto di rete (TCP/IP, NETBIOS, NAMED PIPES)

Il supporto di questi protocolli avverrà tramite un programma di routing, attualmente in sviluppo che permetterà di spedire comandi in rete oppure di convertire in IPC locali del sistema operativo comandi che arrivano attraverso un protocollo di rete (GATEWAY). Ad esempio convertirà comandi che arrivano via NAMED PIPES in DDE e li spedirà al destinatario.

### Importante: Share deve essere in esecuzione

**E' importante che il comando SHARE.EXE del DOS, per gli ambienti operativi WINDOWS e MS/DOS sia in esecuzione, per evitare conflitti nell'utilizzo dei broker FILE e NETF.**

***Approfondimenti:***

Linguaggio NTF

Comando ROUTE

Comando @SEND

Comandi ad oggetti remoti

Definizione di broker

Global Registry

# Variabile @NP

Permette di specificare un PATH differente dove verificare se arriva un messaggio, cioè una transazione complessa, in forma di FILE di testo. Può anche essere un path di rete, ma siccome il parametro deve seguire lo standard NTF, occorre sostituire il carattere \ con \\. A questa variabile fà riferimento il BROKER FILE.

## Modalità d'accesso alla variabile

La variabile è utilizzabile sià in lettura che scrittura.

```
Sintassi:
   @NP=PATH

Esempi:

(In scrittura):
   @NP=\\\\SERVER\\WORKZONE
   @NP=L:\\FILES\\COMANDI

(In lettura):
   @C=@SEND(OBJECT=XX MSG="\B\@(NP)\B")
```

## Note:

**Comunque NTSYSBRK controllerà anche se arrivano comandi nella directory specificata dalla variabile d'ambiente TEMP.**

***Approfondimenti:***

Linguaggio NTF

Canali di input, Tabella dei broker

# *Protocollo di comandi NTF*

## Introduzione protocollo NTF

Possiamo parlare di protocollo invece che di linguaggio in quanto l'obbiettivo di NTF non è quello di essere un linguaggio di programmazione, ma bensì un protocollo per l'interscambio di informazioni o di azioni da compiere, tra oggetti distribuiti, o tra persone e oggetti distribuiti.
Non abbiamo certo la presunzione di inventare qualcosa di nuovo, in quanto esistono già molti protocolli sicuramente più famosi, portenti e standard, come le RPC, il CORBA, le APPC, i socket basati su TCP o UDP.

L'NTF è lo strumento di interscambio di informazioni (o meglio di transazioni) tra applicativi costruiti secondo un modello a componenti, l'architettura NCO, proprietaria di NTG che ha gli stessi obiettivi di semplicità dell'NTF e quindi si pone come alternativo ad altri standard per la realizzazione di applicazioni, come CDE, CORBA, COM.

## L'obiettivo è la semplicità, magari sacrificando qualcos'altro

L'NTF è un linguaggio di tipo transazionale, cioè composto da tante singole transazioni. Si basa sulla stessa filosofia del CORBA, ma con l'intenzione di essere molto più semplice da implementare ed utilizzare, sia dagli applicativi, che dagli utenti finali.

Il nostro obiettivo è proporre qualcosa di semplice a chi deve fare cose semplici e non ha bisogno, o non ha la possibilità per motivi economici o tecnici, di adottare gli standard sopra riportati.

## Transazione atomica NTF

L'unità base ed inscindibile dell'NTF è la transazione. La transazione è così composta:

- **Variabile = Valore**

Tutto il resto del protocollo si basa su questo concetto minimo, ma comune a tutte le transazioni possibili. Il Valore può essere un qualunque valore alfanumerico, che poi verrà interpretato come numerico o alfanumerico a seconda della variabile a cui si vuole assegnare.

## Transazione complessa = gruppo di transazioni

Un blocco di dati (immaginiamo una stringa di testo) contentente più transazioni atomiche NTF, divise da uno spazio o da un invio, costituiscono una transazione complessa. Cioè un insieme di transazioni che devono essere eseguite obbligatoriamente tutte insieme prima che l'applicativo, chiamato anche NCO, passi ad elaborare un altro blocco di transazioni, provenienti da un altro client NTF. Una transazione complessa può essere quindi visto come un programma NTF.

## Variabili

Le variabili possono essere di 2 categorie:

1 **Variabili interne**

2 **Variabili esterne**

### Variabili interne del protocollo NTF (o di un NCO)

Tutte le variabili precedute dal carattere @ sono variabili interne del linguaggio NTF o dell'architettura NCO, quindi il loro uso è riservato. Tutti gli NCO rendono disponibili, in lettura o scrittura, almeno le variabili interne. Ecco la lista di quelle attualmente accettate:


Variabili interne del protocollo NTF (o di un NCO)

### Variabili private di un NCO

Tutte le altre possibili variabili sono private ad ogni NCO. Il nome della variabile segue le regole normali di ogni linguaggio, ed in particolare

- Non possono iniziare con NUMERI o con il carattere @.
- Possono essere utilizzati i caratteri A..Z, 0..9 ed i caratteri speciali "_"," "-", "!", "."
- Non ci sono differenze tra maiuscole e minuscole, l'ntf non considera questa differenza né per le variabili, né per i comandi, invece per i Valori la differenza viene osservata.
- Non ci possono essere spazi all'interno del nome.

## Valori

Il valore di una variabile NTF può essere qualunque carattere del set ASCII esteso, quindi comprendere anche i caratteri da 128 a 255, ma non tra 0 e 31.

- **Attenzione: L'interpretazione di questi caratteri varia da sistema a sistema. L'NTF permette dei costruitti per specificare caratteri con codice ASCII tra 128 e 255, pur utilizzando il set da 32 a 255, cioè l'ASCII standard.**

## Particolarità nella dichiarazione dei valori

- Qualora il contenuto del valore della variabile comprenda degli spazi o caratteri particolari, allora l'intero parametro può essere racchiuso tra "". Tutti i casi seguenti, dove si fà riferimento a caratteri particolari, devono essere specificati tra "".
- Il carattere RETURN (o 13) deve essere specificato con \n.
  Il carattere VIRGOLETTE (o 34 o ") deve essere specificato con \B.
  Il carattere "\" deve essere specificato con \\.
  Il valore di una variabile ntf può essere espanso con \@(XX), dove XX è il nome della variabile NTF., non è importante se viene scritta cone lettere minuscole o maiuscole.
- Per specificare una sequenza di caratteri con valore tra 0..255 (cioè stringa binaria), si utilizza la sintassi \X(hhhh..hh), dove hh è il codice ASC II del carattere in ESADECIMALE.

## Comandi NTF

Un comando è una richiesta di azione ad un NCO. Come per le variabili vi sono dei comandi interni, comuni a tutti i gli NCO e dei comandi specifici per ogni NCO.

I comandi sono associati alla variabile @C. La sintassi di un comando è la seguente:

- **@C=COMANDO(Parametro=Valore .. Parametro="Valore")**

Ogni parametro rappresenta una transazione NTF atomica, con tutte le possibilità che la

sintassi permette. Le parentesi sono richieste solo se si devono inserire dei parametri opzionali o obbligatori. Se non si devono inserire parametri, non sono richieste.

### Sintassi abbreviata comando NTF

Per rendere più simile il linguaggio NTF ad altri linguaggi, abbiamo previsto la possibilità di non utilizzare il prefisso @C= davanti ad un comando. Quindi è accettata anche questa sintassi:

- **COMANDO(Parametro=Valore .. Parametro="Valore")**

Ogni NCO prevede un suo particolare SET di comandi o di variabili, specifici per le funzioni attribuite all'NCO.

### Comandi interni NTF

Come per le variabili interne, esistono dei comandi interni NTF, cioè dei comandi minimi che devono essere supportati da ogni NCO. Questi comandi cominciano con il carattere @, che quindi non può essere utilizzato come carattere iniziale dei comandi specifici di un NCO.

- **@C=@COMANDO_INTERNO(Parametro=Valore .. Parametro="Valore")**

Vediamo in dettaglio i comandi interni del protocollo NTF.

 Comandi interni NTF

### Limiti attuali del protocollo NTF

Ecco alcuni limiti del protocollo NTF:

- **L'architettura NCO attualmente non gestisce blocchi di transazioni NTF (o una singola transazione atomica), più grandi di 16kb.**
- **Mentre viene elaborato un gruppo di transazioni, non ne possono essere elaborate altre. Quelle che arrivano, vengono messe in coda.**

## Protocollo NTB

Si tratta di una versione di NTF minore e compilata, che non richiede un complesso interprete di comandi.

 Protocollo NTB

## NTFCASE: Generatore di transazioni NTF

E' un generatore di stringhe NTF.

 NTFCASE: Editor di transazioni NTF

## Esempi di transazioni NTF

```
Esempi di comandi ntf, interni e specifici di un nco (ntsysbrk):
   @C=EXEC(FILEANME=NOTEPAD.EXE ARGS=AUTOEXEC.BAT)
   @C=DDESEND(APPL=PROGMAN TOPIC=PROGMAN CMD="[CreateGroup(\BNTG
System\B)]")
   @C=DDEPOKE(APPL=EXCEL TOPIC=Sheet1 ITEM=R1C1 VALUE="\@(DT)")
   @C=@STAY

Esempi di variabili interne e specifiche di un nco
   @NP=\\\\ONYA\\CHICAGO
   @DB=1
   BODYTEXT="Prova di testo messaggio, ora: \@(TM)"
   VALUE=150.000
```

## Stile adottabile nello scrivere transazioni NTF

Anche se la maggior parte degli esempi mostreranno delle transazioni con parametri scritte tutte su una linea, queste possono essere spezzate in più righe.

```
Esempio di una transazione NTF suddivisa in più righe:
@C=EXEC(
   FILENAME=NOTEPAD.EXE
   ARGS=AUTOEXEC.BAT)
```

***Approfondimenti:***

 Comandi interni NTF

 Variabili interne del protocollo NTF (o di un NCO)

 Protocollo NTB

# *Variabili NTF*

Tutte le variabili precedute dal carattere @ sono variabili interne del linguaggio ntf o dell'architettura NCO, quindi il loro uso è riservato. Ecco la lista di quelle attualmente accettate:

### Particolarità delle Variabili NTF

- L'accesso alle variabili può essere o solo in lettura o sia in lettura che scrittura. In fase di lettura è consigliabile che l'accesso avvenga tramite stringhe ntf racchiuse tra "".
- L'uso delle variabili è ad espansione. Inserendo il nome di una variabile NTF, con la sintassi \@(Variabile), viene sostituito al costrutto il contenuto della variabile. Il valore della variabile NON viene ulteriormente interpretato dal Parser del linguaggio NTF.
- E' consiglibabile quindi racchiudere il costrutto di espansione di una variabile NTF, tra "" o \B\B a seconda del contesto di utilizzo.

### *Attenzione!! Le variabili NTF hanno un limite....*

- Le variabili NTF hanno un limite nella dimensione (si veda il protocollo NTF per i limiti delle transazioni). Oltre tale dimensione ci sono rischi di troncamento del suo valore.

## Elenco delle variabili interne NTF

`@NP: Net Path`
`@DT: Data odierna`
`@DB: Modalità di Debug`
`@TM: Ora attuale`
`@LM: Risultato ultimo comando`
`@F: Formato dati corrente`
`@CD: Directory corrente`
`@Mx: Variabili buffer`

# Variabile @DT

Ritorna la data odierna in formato americano, indipendentemente dalla nazione.

## Modalità d'accesso alla variabile

E' una variabile a sola lettura.

```
Esempio:
   @C=DDEPOKE(APPL=EXCEL TOPIC=Sheet1 ITEM=R1C1 VALUE="\@(DT)")
```

# *Variabile @DB: Modalità di debug*

Attiva o disattiva la modalità di debugging. Di default è disattiva.
Se la modalità di debugging è attiva, NTSYSBRK salva in un file di LOG nella stessa directory dove si trova NTSYSBRK.EXE, i dati sulle attività principali effettuate dal programma, in particolar modo:

- Avvio di applicazioni
- Invio di comandi DDE
- Invio di messaggi

### Modalità d'accesso alla variabile

La variabile è utilizzabile solo in scrittura.

## Sintassi ed esempi:

```
Sintassi per attivare la modalità di debugging:
   @DB=1

Sintassi per disattivare la modalità di debugging:
   @DB=0
```

## Variabile @TM

Ritorna l'ora attuale in formato americano, indipendentemente dalla nazione.

### Modalità d'accesso alla variabile @TM

La variabile è utilizzabile solo in lettura..

```
Esempio:
   @C=DDEPOKE(APPL=EXCEL TOPIC=Sheet1 ITEM=R1C1 VALUE="\@(TM)")
```

# Variabile @LM

Quando un comando NTF deve ritornare un valore, il suo valore viene ritornato in questa variabile, possibilmente in sintassi NTF. Se un comando non ritorna nessun valore di @LM sarà nullo.
Il valore di @LM può essere memorizzato in una delle variabili buffer interne all'interprete.

Il valore di @LM può essere verificato con il comando @IF, ad esempio per testare il valore di ritorno di un comando.

## Modalità d'accesso alla variabile

E' una variabile a sola lettura.

```
Esempio di memorizzazione di @LM:
   @M1="\@(LM)"

Esempio di verifica del valore di @LM
   @C=@IF(CMP=NOTNULL THEN="@C=@ABORT")
```

***Approfondimenti:***

 Comandi

 Comando @IF

 Variabili NTF Buffer ( @M )

# *Comandi inviabili a NTSYSBRK*

Segue ora l'elenco dei comandi inviabili a ntsysbrk, in sintassi NTF.
Si veda il Linguaggio NTF per una descrizione delle modalità per l'invio di comandi a NTSYSBRK.
Un comando può essere chiamato direttamente o preceduto dalla sintassi opzionale @C=.

## Comandi inviabili a NTSYSBRK

CD
CFG
DDESEND
DDEPOKE
EXEC
KILL
SENDKEY
PAUSE
ROUTE
REGGET
REGPUT

 Comandi interni NTF

***Approfondimenti:***

 Linguaggio NTF

 Comandi ad oggetti remoti

# CD: Cambio della directory e drive corrente

Permette di cambiare la directory e il drive corrente, dove è posizionato il broker. All'avvia il broker è posizionato nella directory dove risiede il programma NTSYSBRK.EXE. Con il comando CD, la directory corrente può cambiare, fino al termine del programma.

In caso di errore, il tipo di errore viene scritto nel file di LOG del broker.

## Sintassi ed esempi

```
Sintassi:
   @C=CD(DRIVE=Drive DIR=Directory)

Ritorno in @LM:
   Directory dove si è posizinati o "" se errore.

Esempi:
@C=CD(DRIVE=T: DIR="\\FREEZONE")
```

***Approfondimenti:***

 Variabile @LM

# CFG: Esame configurazione del sistema

Permette di leggere alcuni dati sulla configurazioni interne della macchina e del sistema operativo. Come risultato ritorna una serie di transazioni CHIAVE=VALORE divise da un carattare di carriage return. Il contenuto viene depositato nella variabile @LM.

## Parametri

Se viene specificato il parametro opzionale RETURN seguito da una chiave, allora verrà ritornato nella variabile @LM solo il valore di quella caratteristica specifica della configurazione, senza essere preceduta dal nome della chiave. In questo modo sarà possibile utilizzare questo valore in altre transazioni NTF seguenti, utilizzando la variabile @LM.

Tabella delle sigle di configurazione interna, riconosciute:

- OSVER: Versione di Windows
- DOSVER: Versione dell'ambiente MS/DOS
- OSDIR: Directory dove si trova il sistema operativo
- WINSYSDIR: Directory system del sistema operativo windows.
- NETUSER: Nome dell'utente attualmente collegato in rete
- NET: Vale "1" se siamo in ambiente di rete locale e se la rete è attiva.

## Sintassi ed esempi

```
Sintassi:
   @C=CFG([RETURN=KEY])

Ritorno il @LM:
   Configurazioni sistema, come transazione complessa NTF

Esempio:
@C=CFG(RETURN=OSVER)
```

***Approfondimenti:***

 Variabile @LM

# DDESEND: Invio di comando a server DDE

Permette di inviare una stringa di comandi ad un Server DDE.

## Sintassi ed esempi

```
Sintassi:
   @C=DDESEND(APPL=Applicazione TOPIC=Topic [ITEM=Item] CMD=Comando)

Ritorno in @LM:
NULL in caso di esito positivo, ERROR o codice dell'errore in caso di
esito negativo.

Esempio:
@C=DDESEND(APPL=PFE TOPIC=System CMD="FileNew()")
```

***Approfondimenti:***

 Comando DDEPOKE

# ROUTE: Instradamento verso oggetti Remoti

## Descrizione del comando ROUTE

La funzione del comando @ROUTE di NTSYSBRK consiste nel passare all'ambiente NTG le informazioni per instradare i messaggi verso oggetti locali o remoti, specificando il canale di comunicazione o BROKER da utilizzare.
Le informazioni verranno salvate in modo persistente, cioè permarranno anche dopo che NTSYSBRK termina, e verranno utilizzate anche da tutti gli altri NCO che verranno eseguiti sulla stessa macchina.

Le informazioni vengono memorizzate nel file NTG.INI in una sezione dedicata per ogni nuovo oggetto che viene registrato, oltre ad una sezione NCO in cui vengono memorizzati i nomi di tutte le connessioni tra oggetti attivabili via routing. I nomi sono unici ed indicando nuove informazioni per un oggetto già esistente, vengono sostituite le impostazioni preesistenti per questo oggetto con quelle nuove.

## Sintassi (standard) del comando ROUTE

```
@C=ROUTE(OBJECT=Nome_Connessione
         NAME=Nome_Oggetto
         BRK=Nome_Broker)
```

### Parametro OBJECT

Nome con cui verrà individuata la connessione all'oggetto remoto. Con connessione si intende un canale di comunicazione aperto con un oggetto remoto attraverso un preciso broker. Il nome della connessione è univoco. Per questo si consiglia, nella scelta del nome della connessione, di associare delle sigle che individuano il broker utilizzato, in quanto è possibile mandare messaggi ad un oggetto remoto attraverso vari broker, ma facendo attenzione di dare ai nomi delle connessioni nomi differenti. Si rimanda agli esempi per chiarire il concetto.

### Parametro NAME

Usato quando il tipo di BROKER richiede un ID. Viene richiesto se si utilizzano i broker DDE, IPX, NETBIOS, TCP/UDP socket, NETF e FILE, con varie differenze operative.

### Parametro BRK

Nome del BROKER di comunicazione con l'oggetto remoto. E' una ID che identifica il broker da utilizzare.

 Tabella dei Broker

## Parametri opzionali o richiesti in certe situazioni

### Parametro PATH

Path dove salvare il file condiviso di estensione .NTF per comunicare messaggi ad un NCO remoto. Può essere anche un path di rete, cioè un PATH composto da \

\nomeserver\dir_condivisa. Viene richiesto nel caso il BROKER scelto sia di tipo NETF.

- Se il BROKER scelto è NETF o FILE, il parametro NAME dovrà essere il nome del file senza estensione (corrispondente all'ID dell'NCO a cui si vogliono mandare le transazioni) Quando NTSYSBRK manderà informazioni all'oggetto che si stà registrando, verrà creato un file di testo con estensione .NTF nella directory specificata con il parametro PATH.
- Se il BROKER scelto è DDE, il parametro NAME dovrà essere il nome dell'Applicazione, corrispondente all'ID dell'NCO a cui si vogliono mandare le transazioni via DDE, usando il topic System e la funzionalità di DDE execute.
- NETBIOS: Da implementare
- IPX: : Da implementare
- TCP/UDP: : Da implementare
- TCP/UDP: : Da implementare

## Esempi del comando ROUTE

```
@C=ROUTE(OBJECT=ONYAFILE BRK=NETF PATH=\\ONYA\TEMP)
@C=ROUTE(OBJECT=NTSQLDDE BRK=DDE NAME=NTSQL)
@C=ROUTE(OBJECT=MSMAILFILE BRK=FILE NAME=MSMAIL)
```

### Descrizione degli esempi:

1 Registra l'oggetto ONYA che sarà in esecuzione sulla macchina ONYA, usando come canale di comunicazione un file condiviso su path di rete.

2 Registra l'oggetto SQL (un'applicazione che invia stringhe in SQL ad Server SQL / ODBC, accettando in input transazioni in ntf) usando come canale di comunicazione il DDE (quindi risiederà sulla stesssa macchina).

3 Registra l'oggetto NTMSMAIL che sarà in esecuzione sulla stessa macchina, usando come canale di comunicazione un file condiviso nella directory identificata dalla variabile d'ambiente TEMP, usata anche da Windows.

***Approfondimenti:***

 Linguaggio NTF

 Comandi ad oggetti remoti

 Configurazione ambiente NTG

 Comando @SEND

 Variabile @F

 Global Registry

# BROKER: Canale di interconnessione tra NCO

E' un canale di trasporto di informazioni attraverso il quale gli oggetti dell'architettura NCO comunicano e si scambiano messaggi in linguaggio NTF.
Alcuni esempi di broker sono:

- Un semplice file condiviso, attraverso il quale processi differenti si scambiano messaggi. Quando un processo stà scrivendo o leggendo sul file condiviso gli altri rimangono in attesa. Il nome del file è associato all'oggetto destinatario (ad esempio NTSYSBRK.NTF per il programma NTSYSBRK).
- Il DDE, il NETDDE, cioè il meccanismo di IPC interno a windows.
- un TCP/UDP socket.
- Un socket NETBIOS.

***Approfondimenti:***

 Linguaggio NTF

 Canali di INPUT

# Comandi ad oggetti remoti

Ecco alcune modalità consigliate per inviare comandi in sintassi NTF ad oggetti remoti NTF, siano essi altri NTSYSBRK in esecuzione su altre macchine, siano essi altri NCO.

## Utilizzo della directory TEMP di un computer remoto.

Visto che c'è su tutti i computer su cui è in esecuzione WINDOWS, è possibile inviare comandi ad un oggetto creando dei file .ntf nella directory di temp, usando "rigorosamente" il programma nxaddntf oppure un altro editor che sia in grado di verificare se il file che si vuole creare è attualmente sotto modifica mentre si tenta di salvarlo (e quindi generare un messaggio di errore per violazione di condivisione).

## Invio di comandi da Shell DOS/Altri ambienti che emulano il DOS

Usando il programma NxADDNTF è possibile inviare via command line o via file esterni comandi ad un nco.

## Esecuzione di Transazioni NTF da File Manager

Si rimanda al paragrafo esplicativo (si veda negli approfondimenti). E' importante aver registrato prima gli nco presenti nel sistema attraverso il comando ROUTE di NTSYSBRK ed in particolar modo aver indicato a quali NCO eventualmente instradare transazioni per nco differenti da NTSYSBRK (usando il parametro FORMAT del comando ROUTE).

## Comandi ROUTE e SEND

E' possibile inviare comandi ad un oggetto remoto, cioè in esecuzione su un'altra macchina seguendo queste modalità:

- Scegliere un broker per comunicare con quell'oggetto
- Registrare l'oggetto + il broker con un nome univoco, indicando le informazioni per il routing delle transazioni, attraverso il comando ROUTE di NTSYSBRK. Si può associare all'oggetto anche un certo formato di dati
- Inviare transazioni a quell'oggetto attraverso il comando @SEND.

***Approfondimenti:***

 Canali di input

 Programma NxADDNTF

 Comando ROUTE

 Comando @SEND

 Variabile @NP

 Esecuzione NTF usando File Manager

# Programma NxADDNTF: Invio comandi NTF da command line

E' un programma che funziona in command line. Permette di mandare una serie di comandi NTF attraverso il canale di input FILE o NETFILE.

Ne esistono varie versioni, a seconda del sistema operativo per cui è stato compilato. Con il pacchetto standard di NTSYSBRK viene fornita la versione funzionante in shell ms/dos (per computer con CPU del tipo almeno 80386sx).

E' la modalità consigliata per l'invio di comandi in queste situazioni:

- da un client MS/DOS, in un ambiente di rete locale, ad esempio in ambiente NETWARE.
- in una shell DOS sotto OS/2 sulla stesssa macchina o su una macchina remota in ambienti di rete NETWARE, OS/2 connect.
- Da una Shell WIN+DOS di WIN95.

Per il funzionamento del programma basta digitare il nome per vedere l'help on line.

## Versioni di NxADDNTF

L'unica versione compresa nel kit standard è quella per ms/dos. Le altre sono include nel kit NTSYS PLUS.

- NDADDNTF: Versione per MS/DOS 16 bit.
- NUADDNTF: Versione per LINUX.
- N3ADDNTF:  Versione per WIN/NT o WIN95 a 32bit.
- NOADDNTF: Versione per OS/2 Warp.

***Approfondimenti:***

 Canali di input

 Comandi da shell MS/DOS

 NTSYS PLUS Add on

# Comandi da Shell ad un NCO

Un uso molto interessante di un NCO, ad esempio NTSYSBRK, consiste nell'eseguire il programma con l'ordine di rimanere residente e poi inviargli comandi sotto varie forme.

```
Esempio:
c:> ntsysbrk @C=@STAY
```

## Invio di comandi via DDE

Se si vuole pilotare un NCO da un altro programma windows, il modo più veloce è l'utilizzo del DDE. Si costruisce una stringa di comandi seguendo la sintassi del linguaggio NTF e la si invia, usando come topic System.

## Invio di comandi da shell DOS

E' possibile inoltre inviare comandi da una shell DOS, ma anche da altri sistemi operativi come UNIX, OS/2, purché possano condividere il file system con la macchina windows dove è in esecuzione l'NCO a cui si vuole inviare il comando.

### La tecnica più semplice, usando il comando ECHO

La tecnica consiste nell'APPENDERE al file ntsysbrk.ntf nella directory contraddistinta dalla variabile d'ambiente TEMP dei comandi in sintassi NTF.

```
Esempio:
ECHO @C=EXEC(FILENAME=WINWORD.EXE) >> %TEMP%ntsysbrk.ntf
```

Attenzione, occorre essere sicuri che ntsysbrk non stia a sua volta adoperando il file (in lettura), altrimenti la shell entrerà in errore di VIOLAZIONE di CONDIVISIONE. E' quindi consigliabile farsi un piccolo programma in C o BASIC che apra in append ed in esclusivo il file ed aggiunga la nuova stringa di comandi NTF. E' inoltre importante che il comando SHARE.EXE del DOS sia attivo.

### Usando NxADDNTF

**Proprio per questo abbiamo incluso nel pacchetto di distribuzione, il programma NxADDNTF.EXE, in versione per vari ambienti operativi che appende comandi ntf ad un file di testo, evitando errori di violazione di condivisione.**

### Con un text editor

Si scrivono le transazioni in formato NTF e poi si salvano nella directory TEMP del computer remoto dove è in esecuzione l'NCO a cui inviare il comando e con il nome dell'NCO + l'estensione .NTF. Attenzione perché una volta recepito dall'nco remoto, il file di testo, viene subito cancellato.

```
Esempio:
C:\TEMP\NTSYSBRK.NTF
```

***Approfondimenti:***

 Canali di Input

 Programma NxADDNTF

 Programma NDREXEC

 Linguaggio NTF

# NDREXEC: Batch per esecuzioni comandi su macchina remota

E' un programma in linguaggio batch, che permette di eseguire applicazioni su una macchina remota, in modo molto semplice, come fosse un client REXEC.
Utilizza il programma NxADDNTF per creare un file dove inviare la transizione NTF.

```
Sintassi:
NDREXEC directory_temp comando [argomenti..]
NDREXEC directory_temp comando ARGFILE file_argomenti
```

### comando:

Deve avere l'estensione e occorre specificare il path preciso, o essere presente in una directory del path.

### directory_temp

Directory temp del computer remoto. Può essere anche nel formato VFAT, cioè \ \COMPUTER\SHARED_DIR.

### argomenti

Uno o più (massimo 6) argomenti, divisi da uno spazio. Attenzione se comprendono caratteri del tipo ' " ', non si può usare questa forma, ma bisogna usare la forma del file di argomenti, per passare questi parametri. Devono essere passati in sintassi ntf (quindi '\' diventa '\\').

### file_argomenti

File di argomenti da dove prendere i parametri. Questa modalità aggira il limite del numero di caratteri specificabili via command line (127) e del non potere il carattere ".

```
Esempi:
  NDREXEC \\BBS\TEMP WINHELP.EXE
  NDREXEC \\BBS\TEMP WINWORD.EXE DOC1.DOC
  NDREXEC C:\TEMP    MSGSEND.EXE ARGFILE DOCS.DAT

FILE DOCS.DAT_:
  "F:\BJohn Mailer\B, T:\BRohn\B, S:\BHello\B B:\BHello Rohn\B"
```

***Approfondimenti:***

Programma NxADDNTF

Comandi da shell MS/DOS

# *NTSYSPLUS: Modulo ADDON per NTSYSBRK*

E' un modulo di ADD ON per NTSYSBRK che ne permette un più facile utilizzo su altre piattaforme. Comprende vari add on, tra cui:

- Versioni per altri ambienti operativi (Linux, Os/2, Win32 Bit) del Front End NxADDNTF e suoi sorgenti.
- Front end grafico per Windows per creare in modo guidato transazioni NTF (NTFCASE).
- La registrazione dà il diritto di ricevere risposta via EMAIL internet ad eventuali domande di approfondimento e di utilizzo di NTSYSBRK e NTSYS PLUS.
- Il diritto di ricevere per 1 anno la nuova versione di NTSYSBRK via Email Internet, oppure via posta, con il solo aggravio delle spese postali (previa richiesta precisa dell'utente, una volta ricevuto l'avviso dell'uscita di una nuova versione).

***Approfondimenti:***

 Modalità di registrazione a NTSYSBRK

 NTFCASE

 NxADDNTF

# *Come contattare NTG*

## How to contact NTG


```
Stefano Petrone
NTG CEO, Sysop Rtt-link BBS.
Fidonet : 2:333/1212.1
Internet: ntg@act.it
BBS     : +39-425-361776 (Rtt-LINK BBS,) (08..24)
        : Fidonet NODE: 2:333/1212
Address : Via A. Toscanini, 39 -  45100  ROVIGO - (ITALY)
```


## Informazioni su NTG e i nostri prodotti

 Informazioni su NTG. Prodotti, Prezzi, Modulo di Registrazione,  ecc.)

 Licenza d'uso dei prodotti NTG

 Procedura per la registrazione o l'acquisto dei prodotti NTG e modulo di registrazione

## Information on NTG and our products

 Updated NTG Informations....(Products, Prices, registration Form, ecc...)

 NTG products user license

 NTG products registration form

# NTFCASE: Editor di transazioni NTF

## In preparazione

Per quanto semplice, certe caratteristiche dell'NTF possono risultare "complicate" da utilizzare quando si scrivono con un editor delle transazioni, sopratutto in caso di utilizzo del comando send, cioè di transazioni reindirizzate.
Per questo motivo insieme ai programmi NTSYSBRK e NxADDNTF per inviare transazioni ntf viene fornito, nel modulo opzionale NTFSYS PLUS, un altro programma, NTFCASE, più propriamente un editor di transazioni con la modalità CASE (cioè guidata), che aiuta a creare le transazioni per i vari NCO creati da NTG.

***Approfondimenti:***

Linguaggio NTF

NTFSYSPLUS Addon

# @SEND: Invio di messaggio ad oggetto remoto

Questo è un comando molto importante. Serve ad inviare comandi ad oggetti NCO remoti. Nella attuale versione dell'architettura NCO, i comandi inviati devono essere in sintassi NTF, in altre versioni potrebbero utilizzare protocolli di comunicazioni secondo vari standard internazionali. Inoltre attualmente il destinatario deve essere un altro oggetto NCO, ma in future versioni potrebbe essere un oggetto costruito secondo altre architetture (RPC, CORBA ad esempio).
Il comando @SEND non interpreta il contenuto del messaggio ma lo invia all'NCO destinatario, utilizzando come BROKER quello specificato nelle informazioni di routing.

## Sintassi ed esempi

```
Sintassi:
   @C=@SEND(OBJECT=Oggetto MSG=Transazioni_NTF)

Ritorno in @LM:
   NULL se non si sono verificati errori, altrimenti la stringa ERR.

Esempi:
@C=@SEND(OBJECT=ONYAFILE MSG="EXEC(FILENAME=\BWINWORD.EXE\B)"
@C=@SEND(OBJECT=ONYAFILE MSG="EXEC(FILENAME=\BWINWORD.EXE\B)"
```

## Note su comando @SEND

- Come si può notare dagli esempi, se il messaggio è racchiuso tra "" e comprende caratteri speciali del linguaggio NTF, questi devono essere convertiti nei corrispondenti simboli preceduti dal carattere di escape. Si veda la sintassi del linguaggio NTF per chiarimenti in proposito.
- **Importante: Viene da sè che non è possibile inviare messaggi ad oggetti che non siano stati precedentemenenti registrati, tramite il comando ROUTE.**

***Approfondimenti:***

 Comando ROUTE

 Comandi ad oggetti remoti

 Canali di Input

 Linguaggio NTF

# File Manager: Esecuzione NTF da File Manager

Possiamo individuare nell'NTF un formato di dati specifico per una applicazione, cioè NTSYSBRK.

Quindi si può associare il formato NTF all'eseguibile ntsysbrk.exe.

Dal File Manager di Windows è possibile quindi eseguire un insieme di transazioni ntf, clickando due volte su un file con estensione .ntf.

E' importante però prima effettuare l'associazione tramite l'Item Associa del Menu File di File Manager dell'estensione .ntf con l'eseguibile ntsysbrk.exe

## Intradamento delle transazioni ad altri nco

Se c'è l'associazione estensione->programma, viene eseguito il broker. In questo caso è importante che la prima transazione del file sia quella che dice al broker il formato dei comandi, in modo che il broker possa instradarli all'NCO opportuno, qualora non siano indirizzati al broker stesso

***Approfondimenti:***

 Avvio di NCO

 Variabile @F

# Variabile @F: Formato dati accettato

Contiene il nome del formato delle transazioni che vengono correntemente accettate dall'oggetto a cui viene richiesto il valore della variabile.

### Ripresa dell'architettura NCO

La convenzione del linguaggio NTF, stabilisce che il nome del formato delle transazioni, corrisponda alla sigla dell'Applicazione o NCO.

In base all'architettura NCO, un'Applicazione, cioè un processo in esecuzione, può contenere al suo interno più oggetti. Questo può essere in grado autonomamente di instradare le transazioni da modificare all'oggetto corrispondente, oppure può essere necessario che sappia a quale NCO le transazioni sono indirizzate.

### Importanza di @F

La variabile @F ha una funzione molto importante, quella di switch di instradamento delle transazioni verso Applicazioni remote che accettano il formato di transazione scelto. E' importante che gli oggetti remoti a cui si fà riferimento siano stati precedentemente registrati nel sistema (in locale) con il comando ROUTE e si sia utilizzato il parametro FORMAT per indicare che quell'oggetto remoto accetta dati nel formato in questione.

L'operazione di modifica della variabile @F deve essere fatta all'inizio di ogni transazione complessa, in quanto alla fine della elaborazione di quest'ultima, viene reimpostato il formato nativo accettato dall'NCO attuale.

### Modalità d'accesso alla variabile

La variabile è utilizzabile sià in lettura che scrittura.

```
Sintassi:
   @F=NOMENCO
```

## Esempi:

```
@F=NTSQL
@C=SQL(SQL="UPDATE CITIES SET POPULATION=POPULATION*1.03;" ID="\@(LM)")
```

In questo esempio si ordina a ntsysbrk di reindirizzare temporaneamente le transazioni che seguono all'oggetto NTSQLDDE, precedentemente registrato con ROUTE per accettare dati in formato NTSQL. La transazione SQL viene mandata all'oggetto in questione che a sua volta si occupa di mandare comandi SQL ad un server SQL.

***Approfondimenti:***

 Linguaggio NTF

 Architettura NCO

 Comando ROUTE

# Architettura NCO: Ntg Component Object

## Introduzione architettura nco

Ce ne sarebbe tanto da dire, ma rimandiamo alla lettura di settore sui component object, come i VBX o i CDX o le DLL di Microsoft, sulla cui scia anche gli nco si pongono.

L'obiettivo è però differente. Mentre i VBX / CDX / DLL possono essere utilizzati solo attraverso linguaggi di programmazione in ambiente windows, gli NCO possono essere comandati anche indipendentemente e senza alcun linguaggio, ma anche attraverso un semplice text editor.

### Possibilità di utilizzo degli nco

Gli nco possono essere utilizzati sia System Manager o utenti finali, sia da programmatori che vogliano utilizzare le funzionalità degli nco all'interno dei loroprogrammi, semplicemendo mandando agli nco, attraverso i canali di comunicazione possibili e più adatti ( BROKER ), stringhe di comandi, secondo lo stesso principio dell'SQL.

 Canali di Input

## Il linguaggio NTF

Per comunicare con gli nco si utilizza il linguaggio NTF, un linguaggio il cui principale obiettivo è di essere semplice, il secondo di essere transazionale.

 Linguaggio NTF

### Prima il BATCH, poi il GUI

Un altro aspetto degli NCO. Tutte le loro funzionalità possono essere utilizzate tramite un linguaggio esterno di programmazione e poi attraverso un'interfaccia utente che però non fà altro che generare comandi nel linguaggio di trasporto delle informaziono, cioè l'NTF.

## Caratteristiche di un NCO

### Applicazione come contenitore di più NCO e porta verso l'esterno

In base all'architettura NCO, un'Applicazione, cioè un processo in esecuzione, può contenere al suo interno più NCO. Questo può essere in grado autonomamente di instradare le transazioni all'NCO corrispondente, oppure può essere necessario che sappia a quale NCO le transazioni sono indirizzate.

 BROKER

### Il formato delle transazioni accettate

Anche se il linguaggio ntf è la base per tutti gli nco, i comandi e le variabili a cui far riferimento (a parte poche eccezioni) sono differenti da nco ad nco, proprio perché ogni nco ha un compito specifico.

## La directory corrente

La directory da dove l'NCO salva o legge i files. Può essere modificata dall'utilizzatore, ma ha certe caratteristiche per permettere l'utilizzo dell'NCO, da parte di più utilizzatori.

Directory corrente nell'architettura NCO

***Approfondimenti:***

Linguaggio NTF

Comandi ad oggetti remoti

Avvio di un NCO

Alias di un NCO

Terminare l'esecuzione di un NCO

## *Directory corrente, nell'architettura NCO*

Per convenzione nell'architettura NCO, durante l'esecuzione di una transazione complessa, è possibile, da parte dell'utilizzatore, modificare la directory di riferimento, modificando la variabile @CD.
Alla fine dell'esecuzione della transazione complessa, viene ripristinata la directory di avvio dell'NCO, cioè il PATH dove si trova l'NCO.

### Attenzione... Questo per oggi non avviene....

Questa possibilità non è ancora operativa, in quanto presuppone di delimiatare le transazioni con degli ID e costruire una piccola macchina virtuale per eseguire le transazioni. Quindi se si cambia la directory corrente, questa rimarrà modificata.

***Approfondimenti:***

 Architettura NCO

## *Alias di NCO*

Un ALIAS di NCO è un nome (univoco) con il quale è riconosciuto sul sistema corrente, un NCO remoto. Quando vengono mandati dei dati ad un ALIAS di NCO, i dati vengono instradati verso un altro oggetto su un sistema remoto (o sulla stesso sistema) attraverso uno specifico BROKER.

# Uscita da un NCO

Per terminare l'esecuzione di un NCO si può operare in un modo interattivo ed uno a comandi.

- Mandando un comando @QUIT attraverso uno dei canali di input possibili
- Chiudendo il task tramite task manager (ctrl+esc). Il task NTSYSBRK e visibile e premendo sul bottone "Fine del Processo" si può terminare il programma.

Non è necessario terminare l'esecuzione di un NCO. Può essere lasciato in esecuzione, fino alla fine dell'esecuzione della sessione del sistema operativo, o in altre parole, fino a che non viene spento il computer.

***Approfondimenti:***

 Canali di Input

 NCO

 Comando @QUIT

# @QUIT: Termina un oggetto in esecuzione

Termina l'esecuzione di un NCO, precedentemente reso residentre tramite il comando @STAY.

## Sintassi ed esempio:

```
@C=@QUIT
```

***Approfondimenti:***

 Comando @STAY

# @STAY: Rimanere residente dopo esecuzione

Comunica ad un NCO di rimanere residente dopo aver eseguito tutte le transazioni pendenti. Come comportamento standard, se un NCO viene eseguito, se non ha transazioni pendenti da elaborare ad esso inviate attraverso linea di comando, questo termina visualizzando la maschera di Help.

Se tra le transazioni passate all'NCO c'è anche il comando @STAY, allora l'NCO rimane attivo anche dopo aver elaborato eventuali transazioni pendenti, aspettando altre transazioni da elaborare tramite i BROKER attivi.

Per terminare un NCO residente, occorre inviargli un comando @QUIT.

## Sintassi ed esempio:

```
@C=@STAY
```

 Comando @QUIT

# Configurazione NTG

Tutte le informazioni di configurazione degli oggetti NTG, compresi quelli del broker sono contenute nel file NTG.INI

Questo file risiede nella stessa directory dove sono installati gli altri NCO. Contiene al suo interno le seguenti informazioni:

## Informazioni di routing

Cioè le informazioni per instradare i messaggi verso oggetti remoti. Tali informazioni vengono modificate o aggiunte tramite il comando ROUTE dell'oggetto NTSYSBRK
Nella sezione [NCO] c'è la lista di tutti gli oggetti registrati, con un nome univoco.
Per ogni oggetto registrato c'è una sezione [ROUTE\OBJ\OGGETTO] che contiene le informazioni di routing per ogni oggetto registrato.

## Configurazioni per ogni NCO

Per ogni NCO, c'è una sezione [NTAPPL\OGGETTO] che contiene le impostazioni di avvio di ogni oggetto, tra le quali.

- Dati dell'utente/organizzazione a cui è registrato il prodotto.
- Chiave di registrazione, quando richiesta.
- Dati sulla release del prodotto.

***Approfondimenti:***

 Comando ROUTE

# Registry: Raccoglitore degli oggetti

E' una componente molto importante dell'architettura NCO. Consiste in un archivio di configurazioni persistenti che vengono mantenute anche dopo la fine del programma. Vengono utilizzate sia da molti comandi dell' architettura NCO sia dall'utente finale che può di sua iniziative, registare o prendere valori dal raccoglitore. Come abbreviazione e per affinità con altre architetture, lo chiameremo anche registry. In certe architetture (Windows) utilizzeremo il formato del registry preesistente in quell'ambiente. E' simile al registry di Windows '95 e Windows NT.
Noi abbiamo preferito mantenere un formato testuale, per rendere più leggibile e modificabile il suo contenuto da parte del System Manager.

Per questa funzionalità sono stati previsti due comandi appositi, REGGET e REGPUT.

## Formato del file che contiene il registry

Dipende dalla piattaforma dove è in esecuzione l'architettura NCO e da eventuali impostazioni dell'utilizzatore. Esiste un formato multipiattaforma che è un file di testo.

## Formato del registry multipiattaforma

Standard generale
[GRUPPO]
OGGETTO = VALORE

Il nome del gruppo o dell'oggetto deve essere tutto maiuscolo e sono accettati solo caratteri alfanumerici del tipo 0..9 e A..Z.

Sottogruppi
[GRUPPO\SOTTOGRUPPO]
OGGETTOSOTTOGRUPPO=VALORE

### Tipi di dati

Valori Booleani (Si/No): Valore Numerico diverso da 0, preferibilmente 1 corrisponde a Si, qualunque altra cosa corrisponde a No. Le stringhe vengono indicate con la sintassi NTF, se contengono spazi devono essere racchiuse tra "".

### Attuali implementazioni del registry

L'attuale versione dell'architettura NCO, tratta il file di registry come un file di testo.

### Prossime implementazioni del registry

La versione attuale è quella multipiattaforma MINIMA, cioè considerata come portabile su tutte le piattaforme. In alcune piattaforme potrebbero essere utilizzati i formati interni dell'ambiente.
In particolare per l'ambiente windows, la prossima implementazione del global registry sara' in formato Windows .reg e utilizzando il registry di Windows NT e Windows 95

***Approfondimenti:***

 Comando REGPUT

 Comando REGGET

 Global Registry

# REGPUT: Scrive oggetti su un file .INI

Scrive un valore in un file di configurazione, nella fattispecie in un file .INI di Windows, cioè all'interno di sezioni. Può essere utilizzato anche per assegnare dei valori al REGISTRY dell'ambiente NTG.

## Sintassi :

```
@C=REGPUT(OBJ=NOME_OGGETTO
   VALUE=VALORE
   REGISTRY=FILE
   TYPE=TIPOFILEREG
   GROUP=GRUPPO)

Ritorno in @LM:
   NULL se non si sono verificati errori, altrimenti codice dell'errore.
```

## Parametri

OBJ: Nome dell'oggetto.

VALUE: Valore da attribuire all'oggetto.

Può essere una stringa contenente qualunque carattere ASCII eccetto il carattere 0.

REGISTRY: Nome del file di configurazione .INI

Non ci deve essere l'estensione in quanto viene appesa di iniziativa da NTSYSBRK e se c'è viene tolta. Se non viene specificato il nome del registry si sottintende che i valori devono essere scritti nel REGISTRY GLOBALE, cioè NTG.INI

TYPEREG: Tipo del file di registry.

INI o REG. Ini e' quello multipiattaforma, REG è quello del registry editor di windows (ancora da implementare). I file .REG possono essere manualmente modificati con il programma REGEDIT.EXE

GROUP: Gruppo dove l'oggetto risiede.

Se non specificato (e se non è specificato nemmeno il file di registry) il valore viene salvato nel sottogruppo NTAPPL\NCO\CFG dove NCO corrisponde al nome dell'oggetto NCO attualmente in esecuzione.

VALUE: Valore da memorizzare come valore dell'oggetto.

***Approfondimenti:***

 Comando REGGET

 Variabile @LM

 Registry

# REGGET: Prende valori da un file .INI

Prende un valore da un registry di configurazione e lo mette nella global return variable.

## Sintassi

```
@C=REGGET(OBJ=NOME_OGGETTO REGISTRY=FILE TYPE=TIPOFILEREG GROUP=GRUPPO)

Ritorno in @LM:
Testo della chiave o NULL se non è stata trovata o in caso di errore.
```

## Parametri

Si rimanda al comando REGPUT per una descrizione dei parametri.

***Approfondimenti:***

 Comando REGPUT

 Variabile @LM

 Registry

# Global Registry: Registro oggetti globale del sistema locale

Contiene gli oggetti standard per il sistema locale dell'architettura NCO e degli oggetti presenti nel sistema, oppure le informazioni di routing per gli oggetti remoti (alias di NCO) memorizzati nel sistema. Attualmente il global registry si chiama NTG.INI e si trova nella stessa directory dell'NCO in questione.
Per avere un unico NTG.INI, si consiglia di installare tutti i prodotti NTG in una UNICA DIRECTORY.

***Per avere un unico NTG.INI, si consiglia di installare tutti i prodotti NTG in una UNICA DIRECTORY.***

## Descrizione dei vari GRUPPI

### Gruppo FORMATS: Instradamento formati delle transazioni

Qualora all'interno di una transazione complessa si cambi la variabile @F e si indichi un certo formato, questo deve essere precedentemente registrato. I dati verranno tutti inviati all'NCO identificato dal formato. Si veda il gruppo ROUTE per le informazioni collegate.

Variabile @F

### Gruppo ROUTE : Informazioni di routing

Sono memorizzate le informazioni per il routing dei pacchetti NTF/NTB tra gli oggetti NCO memorizzati presso nel sistema in esecuzione. Nel gruppo ROUTE, c'è un sottogruppo OBJ e tanti sottogruppi per ogni ALIAS di NCO per cui esistono delle informazioni di routing.

Comando ROUTE

#### Informazioni obbligatorie di routing per ogni oggetto routed

- **NAME : Nome effettivo dell'oggetto sul sistema remoto.**
- **BROKER: Broker utilizzato per l'instradamento dei pacchetti all'oggetto effettivo sul sistema remoto.**

#### Informazioni specifiche di routing dipendenti dal broker

PATH: Path utilizzato dal broker per l'instradamento dai dati all'oggetto effettivo. Viene accettata anche la sintassi estesa \\MACCHINA\PATH.

### *Gruppo NCO: Ntg Component Object registrati nel sistema per il routing*

C'è un oggetto per ogni NCO memorizzato nel sistema per il quale sia previsto un routing delle informazioni.

### *Gruppo NTAPPL: Configurazioni specifiche per ogni applicazione*

Ogni NCO può avere sue configurazioni specifiche che vengono memorizzate e riutilizzate quando il programma viene rieseguito. Ci sono delle configurazioni specifiche di ogni applicazione, ed un sottogruppo CFG dove sono memorizzate le configurazioni globali ma

con un valore specifico per ogni applicazione.

### Oggetti memorizzati nel sottogruppo CFG

- **BRKFILE (1/0): Broker opzionale FILE attivo, oppure no. Dal momento che il broker file funziona in polling, la sua attivazione può rendere più lento il sistema dove è attivo.**
- **BRKNETF (1/0): Broker opzionale NETF (File con path specifico) attivo, oppure no. I motivi sono gli stessi del broker FILE.**
- **BRKNETF (1/0): Broker opzionale NETF (File con path specifico) attivo, oppure no. I motivi sono gli stessi del broker FILE.**
- **USER: Utente a cui l'oggetto è registrato. Utilizzato per il controllo della licenza d'uso.**
- **REGCODE: Chiave di registrazione dell'oggetto assegnata all'USER sopra specificato.**

***Approfondimenti:***

 Architettura NCO

 Alias di NCO

# DDEPOKE: Invio di dato a server DDE

Permette di inviare ad un server DDE, il valore di un item DDE.

## Sintassi ed esempi

```
Sintassi:
@C=DDEPOKE(APPL=Applicazione TOPIC=Topic ITEM=Item VALUE=Valore)

Ritorno in @LM:
NULL in caso di esito positivo, ERROR o codice dell'errore in caso di
esito negativo.

Esempio:
@C=DDEPOKE(APPL=Excel TOPIC=Foglio1 ITEM=R1C1 VALUE=1000)
```

***Approfondimenti:***

 Comando DDESEND

# EXEC - Esecuzione di applicazione esterna

Esegue un'applicazione sulla macchina dove è in esecuzione ntsysbrk. Come comportamento standard, non attende il completamento dell'applicazione e ntsysbrk passa subito ad elaborare la transazione successiva. Inoltre il processo shell viene eseguito in modo iconizzato senza avere il focus.

```
Sintassi:
@C=EXEC(FILENAME=["]Nome_Programma["] [ARGS=["]Command_Line["]])

Ritorno in @LM:
0 se non si è verificato nessun errore, oppure codice dell'errore
verificatosi per lanciare l'applicazione.

Esempio:
   @C=EXEC(FILENAME=N:\\APPLIC\\PFE\\PFE.EXE ARGS=C:\\AUTOEXEC.BAT")
```

## Altri parametri opzionali di EXEC

### Parametro WAIT: Attendi la fine del processo figlio

- Valore <> 0: Attende la fine del processo figlio prima di passare ad elaborare la prossima transazione NTF.
- Valore 0: Non attende la fine del processo figlio prima di passare ad elaborare la prossima transazione NTF.

### Parametro MODE: Modalità di esecuzione del processo figlio

- MIN: Esecuzione dell'applicazione in modalità MIN, cioè ridotta ad icona e senza dare il focus al processo figlio. E' la modalità di default.
- STD: Esecuzione dell'applicazione in modalità normale, dando il focus al processo figlio.
- MAX: Esecuzione dell'appliczione in modalità a pieno schermo, dando il focus al processo figlio.

## Esecuzione di comando della shell DOS

Il comando EXEC, necessita che il programma esterno che viene lanciato esista (un .EXE o un .BAT o un .COM). Se si vuole eseguire un comando della shell MS/DOS (come dir, copy, del, ecc...) non è possibile se non facendolo tramite il programma command.com.
Ecco un esempio esplicativo:

```
Esempio:
   @C=EXEC(FILENAME=COMMAND.COM ARGS="/C DEL FILETEMP.BAK")
```

## Note sul comando EXEC

Nel file di LOG viene descritto l'esito dell'operazione di lancio dell'applicazione esterna.
Occorre sempre specificare anche l'estensione del file di programma esterno.

***Approfondimenti:***


Comando KILL

# KILL: Chiude una applicazione in esecuzione

Permette di chiudere una applicazione in esecuzione specificando il nome della finestra dell'applicazione o parte di esso.

## Sintassi ed esempi

```
Sintassi:
   @C=KILL(APPL=Applicazione)

Ritorno in @LM:
NULL in caso di esito positivo, ERROR o codice dell'errore in caso di
esito negativo.

Esempio:
@C=KILL(APPL="DDE TEST")
```

***Approfondimenti:***

 Comando EXEC

# SENDKEY: Invio di tasti ad applicazione

Questo comando serve a pilotare applicazioni interattive già in esecuzione inviando ad esse delle sequenze di tasti.

## Sintassi ed esempi

```
Sintassi:
@C=SENDKEY(APPL=Applicazione KEY=Tasti)

Esempio:
@C=SENDKEY(APPL=Excel KEY="%{F4}")
```

## Note sul comando

- **Attenzione: questo comando non funziona perfettamente con Windows '95.**
- **Ogni tasto è rappresentato da uno o più caratteri. Se si vuole rappresentare più di un carattere, si possono specificare di seguito. Ad esempio "ABC".**
- **I tasti (+), (^), (%), (~), e le parentesi ( ) non sono usabili direttamente, in quanto vengono utilizzati per scopi particolari. Per specificare uno di questi caratteri, racchiudeteli tra {}. Per esempio, per specificare il tasto +, scrivete {+}.**
- **Le parentesi quadre ([ ]) non hanno un significato particolare per SENDKEY, ma occorre specificare anch'esse tra {}, perché in altre applicazioni per Microsoft Windows, le parentesi quadre hanno significati particolari, in particolar modo usando il DDE. Per mandare i caratteri {}, scrivete {{} e {}}.**

## Tabella dei tasti particolari

| Tasto | Codice | Tasto | Codice |
|---|---|---|---|
| Backspace | {BACKSPACE}, {BS}, {BKSP} | Break | {BREAK} |
| Caps Lock | {CAPSLOCK} | Clear | {CLEAR} |
| Del | {DELETE}, {DEL} | Down Arrow | {DOWN} |
| End | {END} | Enter | {ENTER} |
| Esc | {ESCAPE}, {ESC} | Help | {HELP} |
| Home | {HOME} | Ins | {INSERT} |
| Left Arrow | {LEFT} | Num Lock | {NUMLOCK} |
| Page Down | {PGDN} | Page Up | {PGUP} |
| Print Screen | {PRTSC} | Right Arrow | {RIGHT} |
| Scroll Lock | {SCROLLLOCK} | Tab | {TAB} |
| Up Arrow | {UP} | F1 | {F1} |
| F2 | {F2} | F3 | {F3} |
| F4 | {F4} | F5 | {F5} |
| F6 | {F6} | F7 | {F7} |
| F8 | {F8} | F9 | {F9} |
| F10 | {F10} | F11 | {F11} |
| F12 | {F12} | F13 | {F13} |
| F14 | {F14} | F15 | {F15} |

## Tasti particolari

Per specificare tasti combinati con Shift, Ctrl o Alt keys, far precedere il tasto con uno di questi caratteri:

```
Tasto      Codice
Shift      +
Control    ^
Alt  %
```

Per specificare che i tasti  Shift, Ctrl, e/o Alt devono essere abbassati mentre si preme più di un tasto, racchiudete i tasti da premere tra parentesi. Per esempio, per avere il tasto Shit abbassato mentre si stanno premento i tasi E e C, usare la sintassi "+(EC)".  Per avere Shift abbassato mentre si stà premendo E, seguito da C senza che Shift sia abbassato, usare la sintassi "+EC".
Per specificare una ripetizione di tasti, usare la forma {codice_tasto numero_ripetizioni}. Occorre usare uno spazio divisorio tra il codice del tasto e il numero di ripetizioni. Per esempio, {LEFT 42} ripete la pressioni della freccia a sinistra per 42 volte.

## Note finali

- SENDKEY non può inviare tasti ad applicazioni che non siano per Windows.
- SENDKEY non può mandare il tasto Print Screen (PRTSC) ad ogni applicazione.
- Se qualcuno ha notato similitudini con il comando Sendkeys del Visual Basic per Windows,

confermo in pieno questa senzazione.

# PAUSE: Pausa di n secondi

Mette in pausa l'oggetto NCO per "n" secondi. Durante questa pausa, l'oggetto non verificherà nemmeno se arrivano nuovi dati dai canali di input, ma se arrivano li elaborerà dopo la pausa.

## Sintassi ed esempio:

```
Sintassi:
@C=PAUSE(SECONDS=Secondi)

Esempio:
@C=PAUSE(SECONDS=10)
```

# Comandi interni NTF

Si tratta dei comandi specifici del linguaggio NTF. Questi comandi minimi devono essere implementati da tutti gli NCO.

@ABORT
@CALL
@IF
@FILEGET
@FILEPUT
@SEND
@STAY
@NTF
@FROMNTF
@QUIT

***Approfondimenti:***

- Linguaggio NTF
- Architettura NCO

# @ABORT: Interrruzione esecuzione di transazione complessa

Esce da una transizione complessa NTF (si veda il paragrafo di approfondimento).
Di solito si utilizza in abbinata al comando @IF. Verificandosi la condizione impostata nel comando @IF, se il comando del parametro THEN della IF è @ABORT, la transazione NTF complessa attuale viene abortita.

## Sintassi ed esempi

```
Sintassi:
   @C=@ABORT

Esempio:
   @C=@IF(CMP=NULL THEN=@ABORT)
```

***Approfondimenti:***

- Protocollo NTF
- Comando @IF

# @IF: Esecuzione condizionale di transazione NTF

Effettua un controllo della variabile @LM (risultato ultima operazione) ed in base al valore di questa variabile, esegue oppure no una transazione NTF (atomica o complessa).

## Sintassi

```
Sintassi:
@C=@IF(CMP=Tipo_controllo
    [WITH="Valore di confronto"]
    [THEN="Transazione NTF da eseguire"] )
```

## Parametri

### CMP: Tipo di confronto

Determina il tipo di test da effettuarsi sul valore della variabile @LM. I valori possono essere

***NULL:*** *Esce se @LM non contiene nessun valore*

***NOTNULL:*** *Esce se @LM contiene un valore*

***NOTEQUAL:*** *Esce se @LM non è uguale al parametro WITH*

***NOTEQUALNC:*** *Esce se @LM non è uguale al parametro WITH (senza considereare maiuscole/minuscole)*

***MINUS:*** *Esce se @LM è minore di WITH (se tutti e due sono numeri)*

***MAJOR:*** *Esce se @LM è maggiore di WITH (se tutti e due sono numeri)*

### WITH: Operando di confronto

Da specificarsi solo se richiesto dal tipo di confronto scelto. E' quindi un parametro opzionale

### THEN: Transazione NTF da eseguire in caso di confronto positivo

Se si verifica la condizione richiesta con il parametro CMP, verificando il valore della variabile @LM, viene eseguita la transazione condizionale, specificata con il parametro THEN.

- Attenzione: essendo specificata come parametro, non è possibile utilizzare direttamente i caratteri speciali del linguaggio NTF (") e (\). Occorre prima convertire la transazione, in una stringa NTF. Si veda l'esempio che segue.

## Esempi

```
Esempio:
; Questo esempio prende una transazione complessa NTF da un file
; esterno e la esegue
@C=@FILEGET(FILENAME="C:\\NTF\\SQLSEND.NTF")
@C=@IF(CMP=NULL THEN="@ABORT")
@C=@NTF(MSG="\@(LM)")
@C=@IF(CMP=NOTNULL THEN="\@(LM)")
;
; Se l'esecuzione della transazione caricata dall'esterno
; (SQLSEND.NTF) non ritorna un valore, questa transazione viene
; abortita
```

```
@C=@IF(CMP=NULL THEN="@ABORT")
...ecc.ecc..
```

***Approfondimenti:***

- Comandi NTF
- Comando @NTF

# @NTF: Converte una stringa in sintassi NTF

Questo comando trasforma la stringa passata come parametro in una stringa in sintassi NTF, facendo cioè le conversioni dei caratteri che in NTF sono considerati speciali, come ", \, ecc. in costrutti NTF che li rappresentano ugualmente.

## Utilizzo del comando @NTF

Questo comando viene utilizzato per inserire il risultato di una operazione o di un altro comando, come parametro di un altro comando NTF, qualora il comando precedente non abbia già di sua iniziativa generato un costrutto in sintassi NTF.

Questo si rende necessario perché se non si passasse per questo comando, l'interprete interpreterebbe erroneamente la presenza di caratteri speciali come " e \.
Il comando @NTF fà quindi una elaborazione della stringa convertendo i caratteri speciali che incontra in sequenze di escapes tali che ad una successiva elaborazione, vengano convertire nei corrispondenti caratteri originali.

**Un altro esempio di utilizzo del comando @NTF lo troviamo quando prendiamo da un file esterno un programma NTF.**
**Prima di essere eseguita, una transazione complressa presa da un file esterno con il comando @FILEGET, deve essere processata dal comando @NTF.**

Il risultato del comando viene passato alla variabile @LM.

## Sintassi ed esempi

```
Sintassi:
   @C=@NTF(MSG=Stringa)

Ritorno in @LM:
   Testo convertito in NTF oppure NULL.

Esempio 1:
@C=@NTF(MSG="\@(CD)")


Esempio 2:
@C=REGGET(OBJ=PATHFILE)"
@C=@NTF(MSG="\@(LM)")
@C=REGPUT(REGISTRY="PROVA" OBJ=PROVA MSG="\@(LM)")
```

***Approfondimenti:***

- Linguaggio NTF
- Variabili NTF
- Comando @FROMNTF
- Variabile @LM

# @FROMNTF: Espande una Stringa NTF

Effettua l'operazione inversa del comando @NTF.

## Sintassi ed esempi

```
Sintassi:
   @C=@FROMNTF(MSG=Stringa)

Ritorno in @LM:
   campo MSG in formato normale da NTF, oppure NULL.
```

```
Esempio 1:
@C=@NTF(MSG="\@(LM)")
@C=@FROMNTF(MSG="\@(LM)")
```

***Approfondimenti:***

- Linguaggio NTF
- Comando @NTF
- Variabile @LM

# @CALL: Esegue una sottotransazione NTF

Esegue una transazione complessa NTF esterna che viene inserita in quella in corso di esecuzione.
Per essere più precisi, inserisce le transazioni atomiche della transazione complessa importata, al posto del comando @CALL.
Per questo motivo, la transazione può essere importata da file esterno, oppure costruita all'interno.

**In questo modo l'NTF è in grando di modificare autonomamente e dinamicamente le transazioni complesse in esecuzione.**

## Sintassi

```
@C=@CALL(FILENAME=File_esterno
         MSG=Transazione da eseguire)

Ritorno in @LM:
Non viene modificata ne in ingresso, ne in uscita del comando.
```

### Parametri alternativi

- FILENAME: Importa le transazioni da file esterno
- MSG: La transazione da eseguire viene specificata come parametro.

Non possono essere specificati insieme. Se ciò viene fatto ne verrà preso in considerazione uno solo.

## Esempi

```
Esecuzione di file esterno:
   @C=@CALL(FILENAME="EXTERN.NTF")

Costruzione dinamica di transazioni:
   @M1="WINWORD.EXE"
   @C=@NTF(MSG="@C=EXEC(FILENAME=\B\@(M1)\B)")
   @C=@CALL(MSG="\@(LM)")
```

***Approfondimenti:***

Linguaggio NTF

# @FILEGET: Legge un file esterno

Legge un file di testo esterno e lo carica nella variabile di ritorno @LM. Il file non può essere di dimensione superiore alla dimensione massima accettabile per una transizione complessa NTF (si veda il paragrafo relativo al linguaggio NTF, per spiegazioni in proposito).

## Sintassi ed esempi

```
Sintassi:
   @C=@FILEGET(FILENAME=File [TEXT=1])

Ritorno in @LM:
   File letto come sequenza di caratteri

Esempio:
@C=@FILEGET(FILENAME=DATABASE.DAT)
```

Il parametro opzionale TEXT=1, informa il comando che il file da acquisire contiene dati in formato testo. Di conseguenza verrà convertita la sequenza CR+LF (tipica dell'ambiente PC) nel carattere CR.

***Approfondimenti:***

- Variabile @LM
- Protocollo NTF
- Comando @FILEPUT

# @FILEPUT: Scrive l'ultimo risultato su file esterno

Scrive in un un file di testo esterno e lo carica nella variabile accumulatore @LM.

## Sintassi ed esempi

```
Sintassi:
   @C=@FILEPUT(FILENAME=File [TEXT=1])

Ritorno in @LM:
   Codice dell'errore se si e' verificato un errore o NULL

Esempio:
@C=@FILEPUT(FILENAME=DATABASE.DAT)
```

Il parametro opzionale TEXT=1, informa il comando che il file da acquisire contiene dati in formato testo. Di conseguenza verrà convertita la sequenza CR+LF (tipica dell'ambiente PC) nel carattere CR.

***Approfondimenti:***

- Variabile @LM
- Comando @FILEGET

## Variabili Buffer

Permettono di memorizzare in un vettore di buffer di memoria interna, alcuni valori per potere essere riutilizzati in seguito in altre transazioni NTF, tramite la tecnica della espansione all'interno di una stringa racchiusa tra "".

Le variabili buffer sono 99, da 1 a 99, valori inferiori a 1 o superiori a 99 avranno come ritorno un valore nullo.

### Scrittura in una variabile buffer

```
Esempio:
   @M1="STRINGA MEMORIZZATA"
```

### Lettura da una variabile buffer

```
Esempio:
   @C=SENDKEY(APPL="MICROSOFT WORD-DOCUMENTO1" KEY="\@(M1)")
```

# Variabile @CD: Directory corrente

Ritorna il PATH corrispondente alla directory corrente, della transazione complessa in esecuzione.
Una transazione complessa può modificare la directory corrente, ma alla fine dell'esecuzione della transazione complessa, viene ripristinata la directory corrente di default.

## Modalità d'accesso alla variabile

La variabile è utilizzabile solo in lettura..

```
Esempio:
   @C=@SEND(OBJECT=XX MSG="\B\@(CD)\B")
```

***Approfondimenti:***

- Linguaggio NTF
- Directory corrente nell'architettura NCO

# *Protocollo NTB: NTF compilato*

## Introduzione protocollo NTB

I principi, gli obiettivi e le caratteristiche del protocollo ntb sono gli stessi del linguaggio NTF.
La differenza fondamentale con l'ntf consiste nel fatto che l'ntb è un ntf già compilato utilizzato unicamente tra gli nco per scambiarsi transazioni.
E' esclusa quindi la possibilità di programmazione direttamente da parte dell'utente degli nco da parte dell'utente finale / programmatore attraverso l'ntb. Occorre passare attraverso un livello di compilazione che da una sintassi ntf generi delle sequenze di bytes in ntb.

**In preparazione**

***Approfondimenti:***

- Linguaggio NTF
- Architettura NCO

# NWSENDM.EXE: Per inviare messaggi a programmi Windows

Scopo di questa utility esterna è quello di invare un messaggio ad una applicazione Windows in esecuzione. Questa applicazione può anche essere un NCO in esecuzione sulla stessa macchina dove viene eseguito il messaggio.

### Caratteristiche

Questa utility è un programma per Windows 3.1 o superiori che richiede la presenza dall DLL VBRUN300.DLL.
Funziona solo tramite parametri in command line.
In particolar modo accetta 2 diverse sintassi, sempre di 5 parametri ciascuna. Con NTSYSPLUS daremo la versione a 32bit.

## Sintassi 1 ed esempi

Con la prima sintassi si invia un messaggio ad una applicazione Windows.

```
Sintassi:
   NWSENDM.EXE TYPE PROGRAM wMsg wParam lParam

Esempio:
   NWSENDM.EXE POST NTSYSBRK &12 0 0
   NWSENDM.EXE POST NTSYSBRK 18 0 0
Con questo esempio si ordina al programma NTSYSBRK.EXE in esecuzione di
terminare. Infatti gli viene inviato il messaggio windows WM_QUIT, la
prima volta in forma esadecimale, la seconda in forma decimale.
```

### Parametri:

- TYPE: Tipo di sintassi. Quelle consentite sono: SEND, POST, SENDFILE. La sintassi 1 prevede SEND e POST, che corrisponde a quale delle 2 Api Windows usare per inviare il messaggio, SendMessage o PostMessage.
- PROGRAM: Nome della finestra (o parte del nome) del programma a cui inviare il messaggio.
- wMSG: Messaggio Windows o del tipo WM_USER da inviare alla applicazione. Ogni numero può essere passato in forma decimale o esadecimale (se preceduto da &).
- wParam: Parametro di tipo Word passato al programma.
- lParam: Parametro di tipo Long passato al programma.

## Sintassi 2 ed esempi

Con questa sintassi si può inviare il contenuto di un file allegato al messaggio.

```
Sintassi:
   NWSENDM.EXE SENDFILE PROGRAM wMsg DataFileName lParam

Esempio:
   NWSENDM.EXE SENDFILE NTSYSBRK &465 TEST01.NTF 0
   NWSENDM.EXE POST NTSYSBRK 1125 TEST01.NTF 0
Con questo esempio si invia una transazione complessa NTF al programa
NTSYSBRK.EXE.
```

Parametri:

- TYPE: SENDFILE. Viene usata l'Api SendMessage.
- PROGRAM: Nome della finestra (o parte del nome) del programma a cui inviare il messaggio.
- wMSG: Messaggio Windows o del tipo WM_USER da inviare alla applicazione. Ogni numero può essere passato in forma decimale o esadecimale (se preceduto da &).
- DataFileName: Nome del file da caricare in memoria. La dimensione massima attualmente consentita è di 32768 bytes. L'Handle Globale al blocco di memoria viene passato con il parametro wParam. Il programma destinatario per sapere la dimensione del blocco può utilizzare l'Api GlobalSize.
  Attenzione la dimensione del blocco è arrotondata per eccesso a 64bytes. Il destinatario si deve occupare di copiare il contenuto del blocco di memoria, poiché NWSENDM alla fine della elaborazione lo cancellerà.
- lParam: Parametro di tipo Long passato al programma.

Comandi ad oggetti remoti
Pacchetto NTSYSPLUS
Architettura NCO
Canali di Input

# *Errori e Problematiche d'uso*

## Errori non recuperabili

Qualora avvengano errori non recuperabili, il programma esce, registrando nella directory corrente, un file di nome NTRESULT che contiene un codice di errore.

### Tabella dei codici NTF di errore non recuperabili

- 11: Errore nella sintassi del protocollo NTF.
- 10: File passato come parametro non esiste, o non si riesce ad aprire.

## Errori recuperabili

Tutti gli errori vengono registrati anche nel file NTSYSBRK.LOG che si trova nella stessa directory del programma. Il file di LOG ha lo scopo di dare maggiori informazioni agli utilizzatori, o anche ad NTG per dare un maggior supporto agli utilizzatori, riguardo all'errore riscontrato e al corretto funzionamento delle transazioni.

## Comportamenti anomali

In caso di comportamenti anomali di nostri prodotti, siamo molto interessati a saperlo.

***Approfondimenti:***


- Tabella di tutti i codici di errore
- Come contattare NTG

# *Tabella dei codici di Errore*

Questi sono gli errori interni riportati dall'ambiente di lavoro di NTSYSBRK e di tutti gli NCO. In base al codice di errore che vi viene riportato nel log di lavoro, potete risalire alla tipologia di errore. In molti casi potete risolvere i problemi da soli....

Nel resto dei casi sarà' molto utile ad NTG, sapere il tipo di errore e potete contattarci via EMAIL.

Ritorno al paragrafo Errori

## Tabella degli errori dell'architettura NCO

Errori dell'architettura NCO

## Tabella degli errori Visual Basic

**Qualora non troviate il codice di errore, tra quelli specifici dell'NCO, oppure tra quelli dell'architettura NCO, vi riportiamo i codici di errore specifici del Visual Basic, ambiente di sviluppo con il quale è stata realizzata questa versione dell'architettura NCO.**

| | |
|---|---|
| 6 | Overflow |
| 7 | Out of memory |
| 9 | Subscript out of range |
| 10 | Duplicate definition |
| 11 | Division by zero |
| 13 | Type mismatch |
| 14 | Out of string space |
| 16 | String formula too complex |
| 20 | Resume without error |
| 28 | Out of stack space |
| 48 | Error in loading DLL |
| 49 | Bad DLL calling convention |
| 51 | Internal error |
| 52 | Bad file name or number |
| 53 | File not found |
| 54 | Bad file mode |
| 55 | File already open |
| 57 | Device I/O error |
| 58 | File already exists |
| 61 | Disk full |
| 62 | Input past end of file |
| 63 | Bad record number |
| 64 | Bad file name |
| 67 | Too many files |
| 68 | Device unavailable |
| 70 | Permission denied |
| 71 | Disk not ready |
| 74 | Can't rename with different drive |
| 75 | Path/File access error |

76 Path not found
91 Object variable not set
94 Invalid use of Null
95 Cannot destroy active form instance
281 No More DDE channels
282 No foreign application responded to a DDE initiate
283 Multiple applications responded to a DDE initiate
284 DDE channel locked
285 Foreign application won't perform DDE method or operation
286 Timeout occured while waiting for DDE response
287 User pressed ESC key during DDE operation
288 Destination is busy
289 Data not provided in DDE operation
290 Data in wrong format
291 Foreign application quit
292 DDE conversation closed or changed
293 DDE method invoked with no channel open
294 Invalid DDE Link format
295 Message queue filled; DDE message lost
298 DDE requires DDEML.DLL
320 Can't use character device names in filenames: 'item'
321 Invalid file format
342 Not enough room to allocate control array 'item'
482 Printer error
708 File not found:
710 File already open:
712 Device IO error:
713 File already exists:
716 Disk full:
719 Bad file name:
722 Too many files:
725 Permission denied:
730 Path access error:
731 Path not found:
735 Can't save file to TEMP
757 Can't find Windows Help .EXE file
20000 Can't load Custom Control DLL: 'item'
20001 Can't unload Custom Control DLL; in use
20002 Can't quit at this time

OLE Messages
31001 Out of memory
31003 Can't open Clipboard
31006 Unable to close object
31007 Can't paste
31008 Invalid property value

| | |
|---|---|
| 31009 | Can't copy |
| 31017 | Invalid format |
| 31019 | Source Document is not set |
| 31021 | Invalid Action |
| 31022 | Unable to initialize OLE |
| 31023 | Invalid or unknown Class |
| 31024 | Unable to create link |
| 31026 | Source name is too long |
| 31028 | Object not running |
| 31029 | Dialog already in use |
| 31031 | Invalid source for link |
| 31032 | Unable to create embedded object |
| 31033 | Unable to fetch Link source name |
| 31034 | Invalid Verb index |
| 31035 | Incorrect Clipboard format |
| 31036 | Error saving to file |
| 31037 | Error loading from file |
| 31039 | Unable to access source document |

***Approfondimenti:***

Architettura NCO

## *Errori dell'architettura NCO*

Errori specifici dell'architettura NCO.

Ritorno al paragrafo precedente....

00 ' No Error
01 ' Generic System Error
02 ' Generic Application Error
03 ' Data Type non accepted
04 ' NT Message non accepted by object
05 ' Object Not Found
06 ' File Open Error
07 ' File Read Error
08 ' File Write Error
09 ' Drive or Dir Error
10 ' File do not exist
11 ' Syntax Error
12 ' Link Error
13 ' Range Error
14 ' Memory Error
15 ' Network Error

' DATABASE (20-30)
20 ' Record not found
21 ' Key error
22 ' Query not open. Internal error

## *Uscita da NTSYSBRK*

Si rimanda alla modalità di uscita di un NCO.

***Approfondimenti:***

- Uscita da un NCO
- Comando @QUIT

# *Glossario*

≡ # A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

**#**

@C=

**A**

APPC

**B**

BROKER

**C**

CAE
COM
CORBA

**D**

DSOM

**F**

FOCUS

**G**

GATEWAY

**I**

IPC

**N**

NCO
NTB

**Q**

QBE

**R**

REXEC
RPC

**S**

SQL

**T**

transazione complessa

# *Indice*

**A**



**B**



**C**



**D**



**E**

**P**



**Q**



**R**



**S**



**T**

**V**

# *Vecchie versioni, Novità, Prossime versioni*

## Novità di questa versione

Questa versione è la prima versione ufficiale. Quindi è tutto nuovo. Abbiamo aspettato molto a farla uscire perché avesse già molte delle features previste.

## Storia delle vecchie versioni

Ovviamente, non c'è storia.

## Prossime versioni

- Non è nostra intenzione fare diventare troppo pesante NTSYSBRK. Le prossime versioni avranno lo scopo di migliorare le performance del prodotto, elimiare dei limiti e di aggiungere dei gateway esterni per altri protocolli di rete.
- Con la prossima versione i programmi saranno 2. Uno Freeware che permetterà le attuali possibilità + le nuove aggiunte, una Shareware, dove la registrazione permetterà l'utilizzo di vari protocolli di rete e comprenderà varie utilities di supporto.
- Ovviamente la caratteristica principale della prossima versione, sarà una BUG FIX di questa.
- Ci sarà poi una traduzione completa in inglese ed una completa in italiano.
- Verrà fornito un CASE Interattivo, cioè un'interfaccia utente con la quale sarà possibile creare stringhe NTF per un NCO, non solo per NTSYSBRK.
  Come esempio riportiamo le funzionalità di QBE fornite da tutti i programmi di DBMS per PC, come ACCESS, FOXPRO, per interrogazione dei database in linguaggio SQL.
- La prossima versione del broker sarà in grado di ricevere transazioni anche in un formato non testuale che necessità di essere interpretato, ma anche in un formato binario, già "compilato" specifico per la comunicazione nco<->nco, il formato NTB.

**Sono graditissime nuove idee.....**

## Registrazione prodotto

**Gratis**

La registrazione a questo prodotto è gratuita. Il motivo di questa "cortesia" di NTG consiste nel voler diffondere la sua tecnologia NCO, oltre a voler ricevere dagli interessati opinioni su tale tecnologia, su questo prodotto e su altri prodotti NTG sviluppati secondo la tecnologia NCO.

Quindi potete installarlo come e dove volete, per avere le nuove versioni del prodotto, guardate le informazioni su come contattare NTG.

Solamente nel caso si voglia avere un supporto informativo da NTG e si voglia disporre della nuova versione del prodotto su consegna da parte di NTG attraverso i canali da questa supportati, è richiesta un contributo per il rimborso delle spese.
Tale supporto, viene dato anche nel caso di registrazione (con contributo obbligatorio) al prodotto di add on, NTSYS PLUS.

***Approfondimenti:***

- NTG, Licenza d'uso, Procedura di Registrazione
- NTSYS PLUS Add on

**@C=**

Variabile interna NTF. Serve ad eseguire un comando, rispettando la sintassi NTF composta solo da un insieme di transazioni del tipo VARIABILE=VALORE.

## APPC

Protocollo di IBM di Interprocessing Communication che utilizza come trasporto APPN, estensione di SNA. L'APPC viene usato per realizzazione di applicazioni distribuite che colloquiano con applicazioni Legacy su Mainframe

## BROKER

E' un canale di trasporto di informazioni attraverso il quale gli oggetti dell'architettura NCO comunicano e si scambiano messaggi in linguaggio NTF. Ad esempio un semplice file condiviso, il DDE, il NETDDE, un TCP/UDP socket.

## CAE

Common Application Enviroment, standard per realizzazione di applicazioni, realizzato da X/OPEN, si veda anche XPG4

## COM

Common Object Model, metodologia proprietaria per realizzazione di applicazioni a componenti, basato su OLE2, in via di realizzazione da parte di Microsoft e DEC. Dovrebbe integrarsi con le RPC del DCE.

## CORBA

Standard realizzato da OMG per applicazioni distribuite ad oggetti. Ancora poco usato in quanto nuovissimo.

## DSOM

Distribuite Sistem Object Model, in via di preparazione da parte di IBM, metodologia per realizzazione di applicazioni distribuite ad oggetti, compatibile con CORBA

## FOCUS

Si usa il termine focus per indicare qual'è il programma attivo tra tutti quelli in esecuzione sulla macchina.

## GATEWAY

Un gateway è uno strumento software che converte dati trasportati con un certo protocollo in un altro protocollo (ad esempio da NETBEUI  a IPX).

## IPC

Interprocessing Program Communication. Strumento per l'interscambio di messaggi o dati tra applicazioni in esecuzione contemporaneamente.

## NCO

Sigla di Ntg Object Component. Applicazione NTG che può essere utilizzata autonomamente o può essere utilizzata da altra applicazione o oggetto. Può rimanere residente e quindi diventa una componente aggiuntiva dell'ambiente operativo.

## NTB

E' il corrispondente protocollo "compilato" dell'NTG. Non è in formato testo, ma binario e può essere usato dagli NCO per scambiarsi transazioni più velocemente, in modo che un nco non debba integrare un interprete ntf.

## QBE

Query By Example. Possibilità offerta da molti programmi di creare una stringa di comandi SQL in modo grafico, creando le relazioni in modo guidato con MENU e linee di collegamento fatte con il MOUSE.

## REXEC

Esecuzione di un comando su una macchina remota. Di derivazione UNIX, richiede un deamon sul server e un client per l'esecuzione del comando. E' presente anche in Windows/NT

## RPC

Remote Procedure Call. Il protocollo di veicolamento per applicazioni distribuite in base allo standard DCE di OSF, ed attualmente la metodologia più usata per la realizzazione di applicazioni distribuite, sopratutto in ambiente UNIX.

## SQL

Structured Query Language. Linguaggio di comandi per l'interrogazione o la modifica di Database. Si basa sul concetto di Client/Server, ma è stato implementato anche in modalità non in rete, come con l'ODBC. NTG stà preparando un NCO a riguardo.

## transazione complessa

E' un insieme di più transazioni NTF atomiche, del tipo CHIAVE=VALORE. Tutte le transazioni atomiche che compongono una transazione complessa vengono elaborate, prima che il sistema elabori un'altra transazione complessa, in arrivo da un altro canale (o BROKER) di entrata.