TALK

By: Christopher Lane (Lane@Sumex-Aim.Stanford.Edu)

Uses: Various editor and network protocol modules.

TALK allows users to hold conversations between machines across the Ethernet. TALK uses various *services* (TTY, TEdit and Sketch) and network *protocols* (NS and IP).

TALK FILES

Talk's services and protocols are now in separate files which may be loaded independently:

TALK	The main Talk module.

Services

TTYTALK	Simple text conversation between machines running Lisp, XDE and Viewpoint.
TEDITTALK	Uses TEDIT; allows the full capabilities of the TEdit editor in a conversation.
SKETCHTALK	Uses SKETCH; allows a conversation using the Sketch graphics editor.

Protocols

NSTALK	Uses COURIERSERVE (and optionally NSTALKGAP); allows XNS protocols.
NSTALKGAP	Used by NSTALK if the GAP Courier program has not been defined (by NSCHAT).
IPTALK	Uses TCP and TCPUDP; allows conversations using IP protocols.

Any Talk service can be used with any Talk protocol. The preferred order of loading is:

```
(FILESLOAD TALK TEDITTALK TTYTALK SKETCHTALK NSTALK IPTALK)
```

dropping out those services/protocols you do not use. Order of loading determines which services/protocols are tried first; the files may be loaded in any order to force different priorities.

USING TALK

(TALK *[USER.OR.HOSTNAME SERVICE PROTOCOL]*) [Function]

Starts a TALK session; *USER.OR.HOSTNAME, SERVICE* and *PROTOCOL* are optional. If not supplied, *USER.OR.HOSTNAME* is prompted for (either by menu or typein or both). If *SERVICE* and *PROTOCOL* are not supplied (the usual case) , TALK will figure out which to use based on what is available on the local and remote machines. The supported services and protocols are described below. The service and protocol used are indicated in the title bars of the TALK window.

TALK returns a process handle if the connection is successfully opened; it returns NIL if the user aborts out of the host/user menu and it returns an error message (as a string or list instead of breaking) if it cannot contact the remote host (for whatever reason). TALK can also be invoked from the background menu.

TALK MENU

The menu at the bottom of the TALK window (which is only active while the connection is open) contains the following items:

Disconnect Closes the TALK connection. This is equivalent to closing the TALK window, but leaves the window open in case you want to save and/or hardcopy part or all of the session.

RingBells Rings the bell on the remote workstation (if possible) and flashes the TALK window on the local one to indicate it has done so. This is useful if you have asked a person to hold and want to let them know you have returned.

Message Prompts for and inserts a *canned* message into the TALK stream. Useful if the phone rings and you want to ask the other person to hold with a minimum of time/effort. Messages can be added to the list, see the TALK.USER.MESSAGES variable below. The TALK window must have the keyboard in order to use this item.

ERROR MESSAGES

The TALK function will return one of the following error messages when it fails to start a session:

Host not found! It could not find host address for the host or user name specified.

Can't connect to host! The remote workstation does not have the appropriate server loaded and/or running or does not have TALK loaded.

No answer from TALK service! A connection was made, but no one responded (intentionally or otherwise). A darkened TALK icon is left on the remote screen to log the connection attempt (unless TALK.GAG is non-NIL).

Unknown service type! An unknown type was given as the *SERVICE* argument.

No services available! The *SERVICE* argument was not supplied and it cannot find one.

Unknown protocol! An unknown protocol was given as the *PROTOCOL* argument.

No protocols available! The *PROTOCOL* argument was not supplied and it cannot find one.

Service and protocol errors may indicate additional files need to be loaded**.**

RECEIVING TALK

When your machine is contacted by another via TALK, the following icon will appear on your screen, ringing bells, flashing and showing the time, mode (service and protocol) and (when possible) the caller's identity:

If you button the icon with the left or middle buttons, a TALK session will begin. If you either close the icon or do not button it (it will go *dark* (invert) in about 15 seconds if not buttoned) the TALK connection will be refused. TALK connections are automatically refused if the TALK.GAG flag is non-NIL (settable using the subitem(s) of the TALK item in the background menu)**.** If the machine is in IDLE, TALK will wait twice the normal time out for the user to respond.

If you button a darkened (unanswered) TALK icon, it will try to reconnect you to the caller (after a mouse confirm). If a TALK connection comes in from someone who has already left an unanswered TALK icon on your screen, the icon will be reused.

TALK SERVICES

TEdit

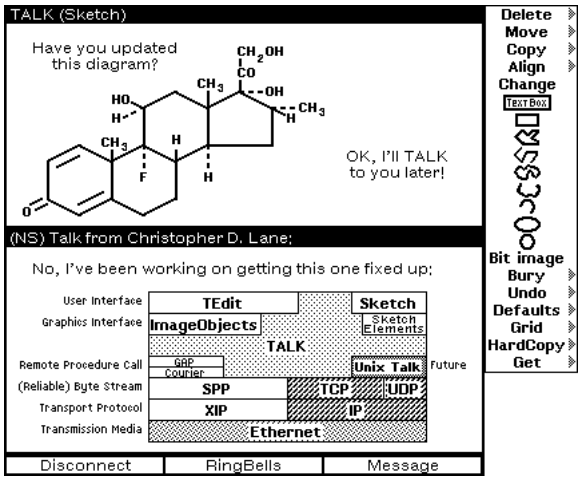
This service allows you to use the full capabilities of TEdit in your conversation, including: correcting mistakes anywhere in the document, changing character and paragraph looks, inserting ImageObjects, etc. Along with keyboard input, mouse selections and the caret are also visible to the remote user. The **GET** and **INCLUDE** commands in the TEdit command menu will load files into both the local and remote TEdit windows, so make sure the files are accessible to both. Similarly for fonts, if your workstation has to load a font from a server, the remote workstation must also have access to the font. Since the remote workstation may also need to load the font, you may experience communication delays. The TEdit service supports NS character codes and most of the 1108 and 1186 function keys.

TTY

This service is similar to the TEdit service except that the only supported feature is *backspace* (but not across lines). TTY is the only service that can talk with the Talk.bcd program in XDE or the TALK application (VPTalk.bcd) in Viewpoint. You *do not* need to know what type of workstation you are contacting when using any of the TALK programs.

Sketch

The Sketch service is built on the Sketch graphics editor:



TALK PROTOCOLS

NS

When NSTALK is loaded, TALK will accept as a host name anything that COURIER.OPEN will accept including an NS address or the name of a workstation registered in a Clearinghouse. Additionally, user names can be used if the address of the user's workstation is registered under the user's name in the Clearinghouse. The following function can be used to register a user and workstation correspondence in the Clearinghouse:

(CH.USER.WORKSTATION *USER WORKSTATION*) [Function]

Sets (or changes) the AddressList Clearinghouse property of *USER* (which must already be a name or alias in the Clearinghouse) to be the address of *WORKSTATION* (an NS address or name). If *WORKSTATION* is NIL, the function removes the AddressList property from *USER*. To use this function, you must be logged in (via `(LOGIN)`) as a System Administrator for *USER*'s domain.

One way to register users would be to go to the individual's workstation, login as the System Administrator and evaluate: `(CH.USER.WORKSTATION 'UserName \MY.NSADDRESS)`
Note that you cannot use the USERNAME function in this example since the (LOGIN) will change it.

NSTALK *does not* require or use NSCHAT, but they do share the Courier program *GAP*. If both NSTALK and the NS CHATSERVER modules are to be loaded, the CHATSERVER should be loaded *first* if possible. NSTALK is designed to allow other types of NSCHAT/GAP servers. The GAP server function determines which function to call using the service type requested (TTY = 5, TEdit = 6, Sketch = 7) and the entries on the association list GAP.SERVICETYPES which has entries of the form (ServiceNumber ServiceName ServerFunction). It is possible to have both NSTALK running and an EXEC server by adding appropriate entries to GAP.SERVICETYPES. If a GAP server already exists when NSTALK is loaded, it is made the default for all unrecognized service types.

Although NSTALK loads the COURIERSERVE LispUsers module you do not have to have a Courier server running to initiate an NS TALK connection, but you *must* have one running in order to receive an NS TALK connection.

IP (Interim)

When IPTALK is loaded, TALK will accept as a host name anything that DODIP.HOSTP will accept, including symbolic and numeric IP addresses. User names can be used by adding them as synonyms for local workstation hosts in the HOSTS.TXT file.

The current TALK IP interface is *only temporary* and will eventually be replaced by one which is compatible (for TTY service) with the TALK program which runs under BSD Unix; at that time, the allowable username format may be expanded to handle user@host. The current IP interface will probably not be compatible with the eventual, Unix-compatible one.

TALK VARIABLES

The following variables can be used to affect TALK's default behavior:

TALK.DEFAULT.REGION = (0 0 500 500) [Variable]

The LEFT and BOTTOM of this region determine where the (initial) TALK icon appears on the screen; the HEIGHT and WIDTH are the combined dimensions of the TALK windows (each uses half the HEIGHT). If this variable is set to NIL, then the icons start at (0 . 0) and the TALK window region is prompted for as needed.

TALK.USER.MESSAGES [Variable]

A list of menu items to put up when the MESSAGES item on the TALK menu is selected. Items on the list should return strings to be put into the TALK stream. If there is an entry of the form (GREETING "message") on this list, it will be printed automatically when a connection is opened.

TALK.GAG = NIL [Variable]

If non-NIL, causes the TALK server to automatically reject any TALK connections.

TALK.ANSWER.WAIT = 15 [Variable]

The number of seconds the TALK icon remains up before closing and aborting the connection.

TALK.HOSTNAMES = NIL [Variable]

A list structure containing hosts TALK has connected to along with the address used.

TALK.SERVICETYPES [Variable]

This list determines which services are tried and in what order. You only need to modify this if you wish to force an order other than the one determined by the order files were loaded or you wish to add or drop a service.

TALK.PROTOCOLTYPES [Variable]

This list determines which protocols are tried and in what order. You only need to modify this if you wish to force an order other than the one determined by the order files were loaded or you wish to add or drop a protocol.

KNOWN PROBLEMS

Talk

- Since TALK uses the Dove/DandeLion sound generator to help announce a connection, on other machines it is difficult for the user to detect connections being made during IDLE.

TTY Talk

- The TTY service cannot backspace beyond the left margin (unlike other implementations).

TEdit Talk

- Sometimes the local and remote TEdit windows will get out of sync as to what the current *looks* are; usually this is not serious.
- Page layout commands have not been implemented for the remote TEdit window; there are probably other commands that do not work either.
- ImageObject specific manipulations to ImageObjects already in the window do not get transmitted to the remote Tedit window.
- Inserting (other than keyboard input) into a pending delete does not echo correctly on the remote Tedit window.
- A large ImageObject inserted into the TALK window may not be seen by the remote user until some text is typed to force the remote window to scroll. The remote user may not see the ImageObject at all if it is larger than his window. These are both true of any TEdit window.
- User scrolling of the TEdit window will not cause scrolling of the remote TEdit window. System scrolling of the window (due to insertions and deletions) will be tracked in the remote window.

Sketch Talk

- When the TALK window is opened, some sketch menus will be created and then replaced. This is due to Sketch not allowing a user to specify both an existing window and an initial menu.
- When text (or a text box) is entered, only the initial character is seen in the remote window until the text is completed and the user buttons some other point in the window.
- Arrow heads do not show up at all on the remote sketch window.
- **Put** of a SKETCHTALK sketch gets into an infinite loop so temporarily you must copy the sketch items to another sketch if you wish to save them on a file.
- If you *sweep* a control point on a box past the other one (like sweeping one corner of a region past the other in RESHAPE), the remote box will not move identically.
- Since there are no functions to programmatically manipulate grouped elements the **Group** and **UnGroup** items have been disabled in the Sketch Talk window.
- For a small number of changes (text fonts, text box brushes and closed wire dashing), the entire remote sketch window is redisplayed to make the change visible.
- Setting the SKETCHINCOLOR flag to a non-NIL value will cause some operations to break.