Richard Acuff
25-Aug-84

Chat Streams

A *chat stream* is a connection between two processes oriented towards terminal service, but not necessarily restricted to that. A chat stream is inherently bi-directional so it is represented by two Interlisp-D streams; one each for input and output. The input stream is considered the primary handle on the connection and is used wherever operations are preformed that are not inherently only input or output. The following operations are available for chat streams (as well as the normal stream operations). In general these operations return true if the operation was successful, NIL if it could not be done:

(**CHAT.SETDISPLAYTYPE** INSTREAM CODE) tells the remote process that a particular type of terminal (designated by the numeric parameter CODE) is being emulated.

(**CHAT.LOGINFO** INSTREAM HOST NAME) determines if LOGIN or ATTACH should be used when logging into HOST with username NAME. This fn is only called by CHAT if it knows how to ATTACH. Returns LOGIN, ATTACH, NIL, or WHERE (when there are multiple jobs to attach to).

(**CHAT.SENDSCREENPARAMS** INSTREAM WIDTH HEIGHT) sends the dimentions of the virtual screen to the remote process.

(**CHAT.FLUSH&WAIT** INSTREAM) insures that the remote process as seen (but not necessarily acted upon) all data written so far.

(**CHAT.OPTIONMENU** INSTREAM) returns a MENU of protocol specific options that can be presented to the user.

Adding a new protocol

To add a new type of chat stream a filter function should be added to the list CHAT.PROTOCOLS which, when given a host name, will return a function. When this function is applied to the host name, it should return a dotted pair (INSTREAM . OUTSTREAM). STREAMPROP should be used to associate the methods for implementing the chat stream operations with INSTREAM. The property names are SETDISPLAYTYPE, LOGINFO, FLUSH&WAIT, SENDSCREENPARAMS, and OPTIONMENU. Only methods for implemented operations need be present.

The function (**ADD.CHAT.MESSAGE** STREAM MSG) is available for getting protocol specific messages to the user. Messages will be typed out on chat window by CHAT, or can be found on the STREAM property MESSAGE by programs. Delimiting white space is added to MSG.

Caveat: CHAT changes the ENDOFSTREAMOP of INSTREAM to return -1 the first time it is called, and then restore the original ENDOFSTREAMOP with the expectation that the -1 will get returned as the value of the BIN that caused the EOS op to be called. While this works for all the protocols I've tried so far, it might not work in general. This kludge is necessary in order to avoid the time-consuming alternative of looping on READP and EOFP. It also did not seem appropriate that the chat stream "interface" implement this kludge, though it might be necessary to resort to that.