XEROX

INFORMATION PRODUCTS GROUP

Systems Development Division

July 22, 1977

To: Tools Group

From: Bruce Parsley, SD/PARC

Subject: Additional Hints about PNRs

Stored: <Parsley>PNRsHints.memo

Here are some "hints" for tool writers concerning Processing Notification Routines (PNRs). The material in this memo will be incorporated into the next release of the TEX spec. You should read the material in that spec in section 5.2 (from about half way down page 6 to the end of the section on page 7). This memo assumes a familiarity with that material (and with some of the material in the Tool User Interface (TUI) and Windows specs).

A tool writer should do something like the following:

> Somewhere in your tool, write one or more PNRs. They are normal Mesa procedures (subroutines) that must have calling sequences as specified by CursorPnrType or PbkPnrType.
>
> Now create a SubwindowObject. The routine CreateWindow makes one subwindow besides the window. The routines CreateSubwindow and EnlinkSubwindow may be used to get more subwindows in a window.
>
> You must create a PNRsObject. This may be done by declaring a (global) variable of that type or by using AllocateHeapNode.
>
> Then you should associate a PNRsObject with each of your subwindows. The way to do this is something like the following: mySubwindow.pnrs ← @myPNRs;
>
> Now you should fill each of the fields of each of your PNRsObjects. The encouraged method is to call one or more of the TUI routines that set up their own PNRs in your subwindows' PNRsObjects. In Release 3.0 of the Tools Environment, Open[Mct/Uas]For[W/Subw]indow are the only such routines.

Eventually there will be similar routines involving menus and selections and possibly other things.

Eventually it should be rare that any tool has its own PNRs, most will use "system" PNRs. Anyway, we'll assume you have your own peculiar PNR that does your own peculiar thing. Then you should do something like the following: myPNRs.keyboardPNR ← MyKeyboardPNR;

Now you should execute a STOP (note that all of the above should be done in your tool's "mainline code" or via global declaractions). Or you may do something like: char ← GetCharP↑[] if you're using a User Action Stream (UAS).

Now your tool just sits around waiting for something exiting to happen, where "exciting" means that the user moves the cursor into (or out of) one of your subwindows, in which case the cursor PNR you have associated with that subwindow will get called, or, while the cursor is in one of your subwindows, the user depresses (or releases) a paddle, button, or key (PBK), in which case the appropriate one (or more if you have overlapping subwindows) of your pbk PNRs will get called.

There are two files in <Tools>TeMesa0300.dm that new tool writers should find helpful. FormTool.mesa provides a "template" for writing tools. TestTool.mesa is an example of an actual, albeit simpleminded, tool. Both of these files should provide hints or examples involving PNRs.

[Aside: I'm not sure into which spec this material should go. Probably the reason why it got mostly overlooked in the first place is that it involves TEX, TUI, and Windows (as with most divided responsibilities, it just got dropped through the cracks). I think that eventually we will want to provide a "Tool Writers' Guide" (in the style of an (informal) programmers' guide). Such a document would be a logical place for this material.]