

Remote Development Interface

The primary goal of the Single Point of Development (SPoD) approach is to provide a development interface that enables software engineers to develop Natural applications for any platform using only the Windows-based graphical user environment of Natural Studio.

The essential new features are:

- **Remote File Manipulation**

In the Natural Studio views, you can manipulate (e.g., move, copy) program objects, regardless of location.

- **Remote Editing**

You can retrieve Natural source files transparently from the target environment, edit them at your workstation and then save them to the target environment. The GUI supported dialogs of the editors provide significant advantages over the character-based editors.

- **Remote Compiling**

You initiate the compilation process from your workstation by issuing commands to the target environment.

- **Remote Debugging**

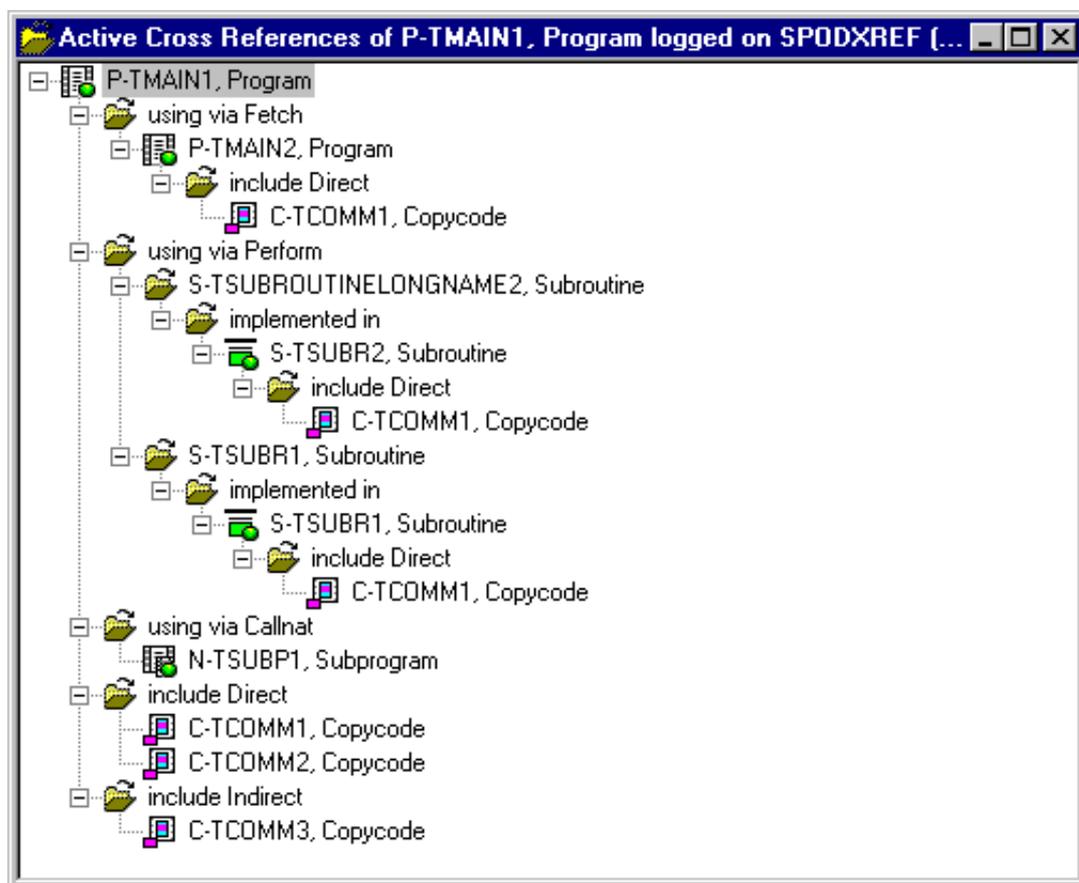
The debugger available with Natural Studio can be used to debug applications which execute in a target environment that can be located on the same system or in Natural mainframe server environments.

- **XREF Information**

An XRef GUI client is available as an optional plug-in unit for Natural Studio. It stores cross-reference information created during CAT or STOW commands in a remote development server file (a system file structured like the FDIC system file). This facility enables you to retrieve and display this essential development and maintenance information conveniently in Natural Studio. This gives you a comfortable way of listing and navigating through hierarchies of referenced and referencing objects, showing relationships (cross-references) between and within program objects. See also XRef Data.

You do not need to have Predict installed on the target environment.

The figure below shows an XRef example screen. For more details and illustrations, refer to First Steps with Natural Single Point of Development.



- **Object Locking**

When you access a remote development server, an object locking mechanism prevents concurrent updates. Locking information is kept in the development server file.

- **Pop-up Terminal Emulation Window**

Maintenance of mainframe applications often involves testing output to terminals. During remote debugging from Natural Studio, a terminal emulation window pops up automatically for that purpose. This terminal emulation is also available when you are using a utility with a character user interface, for example:

```

Terminal Emulation
-----
Session Edit View Help

10:52:42          ***** NATURAL SYSBPM UTILITY *****          2001-08-09
BPNAME          - Main Menu -          Type Local NAT
BPPROP OFF

          Code  Function
          G    General Buffer Pool Statistics
          L    Buffer Pool Load/Locate Statistics
          R    Buffer Pool Fragmentation
          S    Individual Object Statistics
          I    Object Directory Information
          O    Display Object Hexadecimally
          D    Delete Object from Buffer Pool
          B    Blacklist Maintenance
          P    Preload List Maintenance
          F    Internal Function Usage

          Code .. _  Library ... * _____
                   Object .... * _____
                   DBID ..... 0    FNR .. 0 _____

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
          Help      Exit  Last      Flip                                Canc
4AÛ                                                    17,021

```

In summary, the benefits of remote development and maintenance are:

- Better control over Natural development tasks
- Increased productivity through a powerful graphical work area
- Lower costs for software development, maintenance and administration
- Greater job satisfaction for software engineers