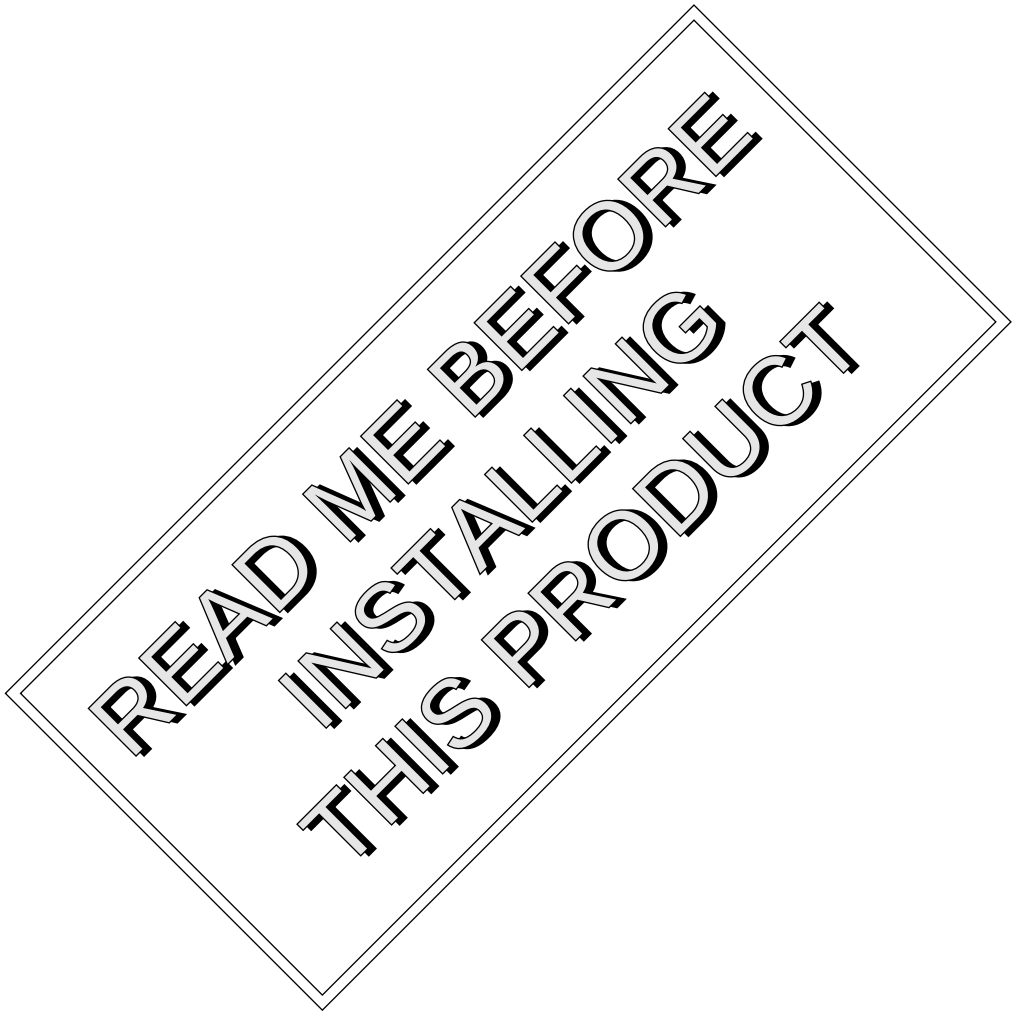
NightView
Version 5.1 Release Notes

January 2000

0890395-5.1



CONCURRENT
COMPUTER
CORPORATION

Copyright

Disclaimer

The information contained in this document is subject to change without notice. Concurrent Computer Corporation has taken efforts to remove errors from this document, however, Concurrent Computer Corporation's only liability regarding errors that may still exist is to correct said errors upon their being made known to Concurrent Computer Corporation.

License

Trademark Acknowledgments

Contents

1.0. Introduction

NightView™ is a general-purpose, source-level debugger for Ada, C, C++, and Fortran. NightView can be used to debug multiple processes on the local system or on different hosts.

NightView is supported on systems based on the PowerPC™ running PowerMAX OS™.

NightView 5.1 has these major changes:

- NightView now uses HyperHelp™ for online help.
- NightView 5.1 has improved support for Ada and C++ inline subprograms.
- NightView supports command history. You can retrieve earlier NightView commands, edit them, and resubmit them.

NOTE

Installation procedures for this product are slightly different from those for other Concurrent products. Please read *System Installation* on page 4 of these release notes before attempting to install this product, and carefully follow the installation instructions.

In these release notes, chapter and section references are to the *NightView User's Guide*, unless stated otherwise.

2.0. Documentation

Table 2-1 lists the NightView 5.1 documentation available from Concurrent.

Table 2-1. NightView Version 5.1 Documentation

Manual Name	Pub. Number
NightView User's Guide	0890395-200
NightView Version 5.1 Release Notes	0890395-5.1
NightView Pocket Reference	0890475-030

Copies of the Concurrent documentation can be ordered by contacting the Concurrent Software Support Center. The toll-free number for calls within the continental United States is 1-800-245-6453. For calls outside the continental United States, the number is 1-954-973-5354.

Additionally, the *NightView User's Guide* and *NightView 5.1 Release Notes* are available online by using the X Window System™ utility, **nhelp**.

Further, the *NightView User's Guide* and *NightView 5.1 Release Notes* are also available on Concurrent Computer Corporation's web site at www.ccur.com.

3.0. Prerequisites

Prerequisites for NightView Version 5.1 are as follows:

3.1. Software

- PowerMAX OS Version 4.2 or later
- X Window System (X11 Version 6.3 or later)
- Normally, NightView uses 2-3 lightweight processes (LWP's). On some systems, the number is much higher. This problem is fixed by PowerMAX OS release 4.2 patches base-004, base-005, and base-006. This problem is also fixed in PowerMAX OS release 4.3.
- An X™ server is required if you are using NightView's graphical user interface.
- Élan License Manager™ 5.0.1 or later (See *System Installation* on page 4 for details)

For more information about configuring your system to support NightView, see Appendix B, "System Resource Requirements," in the *NightView User's Guide*.

3.2. Hardware

- Any system based on the PowerPC running PowerMAX OS

4.0. System Installation

The NightView product is installed as standard PowerMAX OS software packages and utilizes the standard PowerMAX OS product installation mechanism, **pkgadd** (see **pkgadd(1)**).

The package names are **NightView** and **Nviewp**. These names are case-sensitive.

Please refer to the “Installing Add-on Software” chapter in the *System Administration Volume I* (0890429) manual and the *PowerMAX OS Release Notes* for instructions on software installation.

NightView may be installed in either the root directory or elsewhere. When you run **pkgadd(1M)** to install NightView, you are prompted to enter the name of the directory for installation. If you want to install in the root directory, just press the <return> key at the prompt. Otherwise, enter the name of the directory where you want NightView installed. If this directory does not exist, the installation procedures attempt to create it for you.

NOTE

NightView consists of multiple packages. The **NightView** package must be installed on systems on which you want to run the NightView user interface. The **Nviewp** package must be installed on systems on which the user programs you want to debug will be running. Most customers will want both packages on the same system.

If the NightView package is installed before the **Nviewp** package, you will get a warning about the **Nviewp** package not being installed.

NightView **must** be run with the Élan License Manager. Follow the steps in the "Obtaining Licenses" section of the *Élan License Manager Release Notes* (0891055); the *feature alias* is **NightView**. If you are debugging Ada, then you need an additional license. The feature alias to debug Ada is **NightView_Ada**. If you are not already running the Élan License Manager, if you do not have a copy of the *Élan License Manager Release Notes*, or if you need a license key, contact Concurrent Software Distribution at 1-800-666-5405 (or 1-954-283-1836 outside the continental United States).

If your site has multiple license servers, and you need to indicate a server on a particular system, you can set the environment variable POWERWORKS_ELMHOST to the name of the server’s system before invoking NightView. For more information, see the *Élan License Manager Release Notes* (0891055).

5.0. Overview of NightView 5.1

5.1. Enhancements

5.1.1. Online Help

NightView now uses HyperHelp for online help. You can access the *NightView User's Guide* through NightView or through **nhelp(1)**. In the graphical user interface, help is available from the Help menu or with the **help** command. In the command-line interface or the simple full-screen interface, help is available only for error messages--the *User's Guide* is not available in those interfaces. See "help" in Chapter 7.

5.1.2. Inline Subprograms

NightView 5.1 has improved support for inline subprograms in Ada and C++. Generally, inline subprograms look like non-inline subprograms.

- Inline subprograms appear to have stack frames, so you can use **up**, **down** and **backtrace** with them. See "up", "down" and "backtrace" in Chapter 7.
- **next**, **step** and **finish** work with inline subprograms. See "next", "step" and "finish" in Chapter 7.
- The **interest** command applies to inline subprograms. If you set an interest level for an inline subprogram, it applies to all instances of that subprogram. The **interest** command has a new keyword, **inline**, to set or query the interest level for all inline subprograms that do not have an explicit level. See "interest" in Chapter 7.

There are some limitations to the inline subprogram support.

- You can call inline subprograms directly in debugger expressions only if there is an out-of-line instance of that subprogram. The subprograms you call directly may call inline or non-inline subprograms.
- For C++, NightView considers inline subprograms to be different if they are in a different compilation unit. Therefore, setting an interest level for an inline subprogram, or setting an eventpoint in an inline subprogram, applies only to instances of that subprogram within the same compilation unit.

5.1.3. Command History

NightView 5.1 supports command-line history in the graphical user interface and the simple full-screen interface.

For the graphical user interface, commands are entered in a combo box. You can use down-arrow to cycle through earlier commands, or use CTRL-down arrow or click on the triangle ("down arrow") to bring up the scrollable list of prior commands and select one. Once a command has been retrieved, you can edit it with the normal Motif™ keys, such as left and right arrow, Home, End, Backspace, Delete, and CTRL-Delete.

In the simple full-screen interface, editing and command history is similar to the Korn shell. See **ksh(1)**. NightView considers the environment variables VISUAL and EDITOR, to choose whether to

be in vi or emacs mode. There is also a command: **`set-editor`** *mode*, where *mode* is one of `emacs`, `gmacs` or `vi`. See "Command History" in Chapter 3.

5.1.4. Reserving Memory

NightView uses some areas of memory in your process address space for things such as eventpoints and to evaluate some expressions. The **`mreserve`** command lets you reserve areas of memory in your process address space so that NightView will not use them. Most users will not need this. See "mreserve" in Chapter 7.

5.2. Changes in This Release

5.2.1. Ada Raise Occurrence

NightView now supports Ada exceptions raised via the `ada.exceptions` mechanism as well as via the `raise` statement. See "handle" in Chapter 7.

5.2.2. Ada Exceptions and Restart

Release 5.1 changes the behavior of **`handle/exception`** and **`handle/unhandled_exception`** with respect to restart. The state of those commands is always explicitly put into the restart information by a checkpoint, so that they will override any **`on program`** settings. See "handle" in Chapter 7. See "Restart Information" in Chapter 3.

5.2.3. Ada Array Subscripts

Release 5.1 fixes problems with expressions that subscript an Ada array with a small enumeration.

5.2.4. Comparing Ada Strings

Release 5.1 fixes problems with expressions that compare Ada strings where one string is a prefix of the other string.

5.2.5. Ada Declare Blocks

You can now set eventpoints on subprograms within Ada declare blocks if the current frame is within the declare block. See "Current Frame" in Chapter 3.

5.2.6. Returning a Reference

NightView now works correctly with C++ expressions that involve a function that returns a reference.

5.2.7. License Manager

Earlier releases of NightView would sometimes fail to release a license after you quit NightView. That is fixed in release 5.1. See "Invoking NightView" in Chapter 6.

5.2.8. Process Memory Use

Release 5.1 changes the way NightView uses memory in the user process.

- The patch code for individual eventpoints is smaller than before.
- Earlier releases of NightView would use multiple "text" areas. Release 5.1 uses multiple "event-point" areas, and a single "text" area. The eventpoint areas contain only code that needs to be near certain sections of your program. All other code goes into the text area. The default size of the eventpoint areas is smaller than the default size of the text areas was in earlier releases. See the appendix titled "Implementation Overview". The **`set-patch-area-size`** command has a new keyword, **`eventpoint`**. See "set-patch-area-size" in Chapter 7.
- **`info memory`** now reports how much space is left in each patch area. With the **`/verbose`** option, **`info memory`** gives information about each block within each patch area. Most users

will not need the verbose information. The output format for **`info memory`** now includes the end address of each region. See "info memory" in Chapter 7.

5.2.9. NightBench

If you debug a program from NightBench, and NightView's windows are all iconified, NightView de-iconifies a window.

5.2.10. Floating-point NAN

Release 5.1 fixes problems with displaying floating-point NAN (not a number).

5.2.11. Busy Feedback

The simple full-screen interface now has busy feedback similar to the busy feedback in the graphical user interface.

6.0. Cautions

6.1. Ada Tasks, Threads and Lightweight Processes

NightView 5.1 does not yet have direct support for Ada tasks, threads or lightweight processes. However, the **`select-context`** command provides some support. See "select-context" in Chapter 7.

6.2. Ada Support

Many Ada features are supported, but some are not. See "Ada Expressions" in Chapter 3.

6.3. Breakpoint Command Streams

The current implementation of breakpoint command streams does not include protection against concurrent access of data structures internal to the debugger. Avoid referencing other processes within breakpoint command streams. In practice, this is rarely a problem.

6.4. Fortran 77 Arrays

There is currently a problem with Fortran 77 arrays passed as formal arguments. The compiler generates a temporary variable to optimize references to the array. This temporary variable is then used in place of the original array argument. If the original array argument is not used later in the routine, then the lifetime of the argument is very short and its value is discarded. If you try to use the debugger to reference the argument later in the routine, the results are undefined. You can do two things to avoid running into this problem:

- Use the array later in the routine. For example, pass the array as an argument to another routine. This extends the lifetime of the original argument.
- Use the debugger to look at the array at the first executable line of the routine. The value of the original argument probably still exists at this point.

6.5. Scroll Bars

Some users have had a problem using the scroll bars in the graphical user interface. When the up-arrow button on a scroll bar is clicked, the window scrolls the length of the entire window contents, rather than just one line. This problem is caused by an inappropriate X resource, which has apparently been copied from one user to another in much the same way as **`$HOME/.profile`** files are often copied:

```
*XmScrollBar*translations:   <Btn1Down>:   Select()
```

You should remove this resource from your **`$HOME/.Xdefaults`** or whatever other means you use for setting X resources. This resource affects any Motif application, not just NightView.

6.6. Accessing Variables While in Prologue of Subprogram

Because the compilers generate line-number information for code that comprises the prologue of a subprogram, you may inadvertently set an eventpoint or stop your program at such a location. The prologue is the code that the compiler generates to set up the execution environment for a subprogram; until that code completes execution, the environment is incomplete. Attempts to access variables while in the context of the prologue may result in errors from NightView or may result in erroneous values.

If you suspect such a problem, you can determine whether you are in the prologue by doing the following:

- Stop the program at the point where the referencing problem occurs.
- Enter the command **`info frame`**.
- If the output from this command says something like "Could not obtain frame description information", then you are probably stopped in the prologue.

To try and avoid this problem, do not set an eventpoint on the lines containing a subprogram heading. If you want to set the eventpoint at the beginning of the subprogram, use the *unit_name* (for Ada) or *function_name* form of location specifier, rather than a *file_name* and *line_number*. See "Location Specifiers" in Chapter 7.

6.7. Shared Objects

For core files, NightView cannot always find correct stack description information (tdesc) for shared objects. This problem occurs when the shared objects are built without position-independent code. This problem can cause NightView to display the wrong results for any routine other than the most recently called routine. For example, a **`backtrace`** command may print the wrong information, or an **`up`** command may fail. This is true for any release of NightView, not just 5.1. This is not a problem for running processes, only for core files.

We recommend that you build shared objects with position-independent code if you plan to debug programs that use them. For C and Fortran, use the **`-Zpic`** option. See **`cc(1)`** or **`f77(1)`**. For Ada, use **`-sm shared`** or **`-sm both`**. For HAPSE Ada, see "Shared Libraries and Shared-Objects" in the *HAPSE Reference Manual* (0890288). For MAXAda, see "Compile Options" and "Shared Objects" in the *MAXAda Reference Manual* (0890516).

7.0. Direct Software Support

Software support is available from a central source. If you need assistance or information about your system, please contact the Concurrent Software Support Center at our toll free number (1-800-245-6453). Our customers outside the continental United States can contact us directly at 1-954-973-5354. The Software Support Center operates Monday through Friday from 8 a.m. to 7 p.m., Eastern Standard Time.

Calling the Software Support Center gives you immediate access to a broad range of skilled personnel and guarantees you a prompt response from the person most qualified to assist you. If you have a question requiring on-site assistance or consultation, the Software Support Center staff will arrange for a field analyst to return your call and schedule a visit.