Chapter 4 Software Development Tools

Sun has several products to assist in software development. This chapter gives an overview of:

- *Programming Environment Support Tools*: A set of programming utilities bundled with SunOS.
- *Network Software Environment (NSE)*: A comprehensive software development platform, sold separately.
- *SunTrac™ Project Management System*: A project management system using sophisticated analysis with the advantage of networked computing technology, sold separately.

4.1 Programming Environment Support Tools

Programming Environment Support Tools (formerly known as SunPro) are a set of programming utilities that come bundled with SunOS. These utilities form a productive programming environment for developers of product release software. Programming Environment Support Tools include: shell programming capabilities, editors, floating-point operations, preprocessors, C compiler, debuggers, performance analysis utilities, and source-code control utilities. The following outline gives an overview of the capabilities available, and pointers to detailed documentation on each area.

4.1.1 Organizing a Programming Environment

Shell programming capabilities are UNIX features that customize your working environment. These features include: setup files (.login and .cshrc), root menu and SunView customization, and the use of shells and shell facilities (online help; command aliases; I/O redirection; pipelines; installing; running and debugging shell scripts; grep; nm; and strip).

4.1.2 Editing Code

Sun's version of the UNIX operating system provides several editors for general use on text files: the *ed* and *ex* line editors, and the *vi* and *textedit* screen editors. In addition, other editors (such as Emacs) are available from third-party vendors. SunOS also provides an indent facility. *Indent* is a C program source-code formatter. It reformats a C program according to an extensive set of options.

4.1.3 Organizing and Storing Code with SCCS

The Software Code Control System (SCCS) is a software management tool that helps coordinate a team of programmers working on a large software development project. SCCS serves as a custodian of program source files, determining who may have access to files and who may edit them. SCCS maintains when a file is edited, and keeps track of the revision levels of each file. A check-out system prevents the possibility of multiple uncoordinated updates. SCCS commands include *create* (create a file in SCCS), *get* (get a copy of an SCCS file on which to work), *sccs* edit (edit the file copy), and *sccs* delta (incorporate changes to form the new version of the file).

4.1.4 *make:* Keeping Files Up to Date

The *make* utility assists programmers in recompiling code thoroughly and efficiently. A user-defined makefile lists all dependent modules that are required to construct a target program. When *make* is invoked, it looks at the UNIX timestamp placed on each dependent file listed in the makefile to determine when it was last revised. *make* then recompiles only modules that have a timestamp later than the last target recompile. *make* is efficient and thorough, in that only modules that have been altered are recompiled. Sun's implementation of *make* contains many enhancements. Most notable are hidden dependency checking and command dependency checking.

4.1.5 Debugging Code

Sun offers two types of debuggers: *dbx*, a source-level debugger, and *adb*, an assembly-level interactive debugger.

Sun's *dbx* is an extended and debugged version of the Berkeley 4.2BSD *dbx*. Enhancements that Sun has added include robustness; debugging capabilities for Pascal, Modula-2 , C, and FORTRAN; faster initialization; source code searching; expanded evaluation of C operators; improved stack-walking facilities; arbitrary-process debugging; multiple-process debugging; and kernel debugging.

Dbxtool is a graphic interface to dbx that eases and speeds development on Suns

A window/mouse-based version of *dbx*, called *dbxtool*, has become the debugger of choice on Sun systems. Using only the mouse pointing device, users can run the program, set and clear breakpoints in the source code, display the values of variables, display a call stack trace, single-step a program, and continue execution from a stopped condition. In addition to the mouse-selectable functions, *dbxtool* provides a window in which you can type any of the normal *dbx* commands. Although *dbxtool* is not a true integrated programming environment, it exhibits the major features of one: you can edit, compile, link, and debug a program within *dbxtool*.

adb is an assembly-level interactive debugger. *adb* is more difficult to use, but it can do some things that *dbx* cannot, such as debug programs not compiled with the -g symbol table option and patch code in memory or in a file. *adb* is a good general tool for interactively examining binary files.

Note: Another source-level debugger commonly found on UNIX systems is *sdb* from Bell Laboratories. Sun supplies *dbx* rather than *sdb* because the former has better capabilities for controlling execution and for displaying complex data-structure elements. For those already familiar with *sdb*, *dbx* provides a command aliasing facility that lets users alter its syntax to resemble that of *sdb*.

4.1.6 Performance Analysis Utilities

Performance analysis tools are of two major types: those that monitor the performance of an individual program, and those that monitor aspects of system performance as a whole.

Sun's operating system provides four major tools that examine the performance of an individual program:

- *time* Monitors execution time.
- *size* Displays the size of different sections of the program.
- *prof* Constructs a profile of time spent per routine, subroutine call frequency, and average time spent in each routine per call. (*gprof* is an enhanced version of *prof* that produces a call graph that identifies the runtime relationships between routines.)
- *tcov* Produces reports that show which parts of the program were exercised in a test run and which parts remain to be tested. (*pxp* is the Pascal equivalent of *tcov*.)

For system analysis, Sun provides utilities that gather statistics on system performance:

- *perfmeter* Determines whether the program is CPU-bound.
- *perfmon* Provides statistical display of CPU use.
- *vmstat* Reports virtual memory statistics.
- *iostat* Reports I/O statistics.
- *netstat* Reports network memory management and interface statistics.
- *mfsstat* Reports kernel-network interface statistics.

4.1.7 Source-Code Control within NFS

Sun's Network File System (NFS) makes any number of remote filesystems appear to be mounted locally. After file systems are exported and mounted with NFS, a remote file can be accessed with standard UNIX I/O calls, regardless of its location in the network. The most important advantage of NFS for programmers is the ability to import and export files transparently across the network, provide immediate access to any file of interest (provided permissions have been granted), and conserve local disk memory. For a team of developers to take advantage of NFS, the use of network groups and the use of SCCS in NFS should be understood.

Network groups were invented for use with the Yellow Pages to simplify the permission checking associated with remote mounts, logins, and shells. Network groups classify machines and users on the network who are allowed access to files and operations on other machines. For remote mounts, the information in network groups is used to classify machines; for remote logins and remote shells, it is used to classify users.

Because NFS does not lock files, SCCS is commonly used to control access to source code in the NFS environment. To control parallel development, when a file is checked out for editing on each machine, SCCS creates a local lock file so that no one else can edit the file until it is returned. In sequential development, a common way to share the SCCS directory is to form a symbolic link between the SCCS directories. When two programmers use the same SCCS files for their source, they eliminate the need for merging because their work proceeds sequentially - they edit the same file at the same time.

For more information, the following documents are recommended:

- *The UNIX System: A Sun Technical Report*
- *SunPro: The Sun Programming Environment (A Sun Technical Report)*

4.2 Network Software Environment (NSE)

NSE is a comprehensive software development platform that improves software quality, development productivity, and programming performance

As software engineering projects become increasingly complex, the adoption of a systematic approach to software planning, development, and tracking is imperative. Sun's Network Software Environment (NSE) is a comprehensive software development platform that improves software quality, development productivity, and programming performance.

Project administration and coordination difficulties pervade the entire software lifecycle. The NSE supports the various development objects used during the different phases of software development. These objects can include source code and associated program files, project requirements, design documents and drawings, test data and drivers, schedules, budgets, and staffing plans. The NSE Version Control System records and maintains the versions and histories of all development objects.

NSE developers work in individual work spaces called environments. Environments provide isolation and stability, maintaining object naming consistency and a hierarchy of development objects and associated tools. Developers can make changes to development objects without effecting each other's work. Because the NSE allows developers to work concurrently on the same object, they must eventually merge their changes. Ordinarily, conflicts arise when developers merge their changes together into a single version of a program, but NSE detects all program integration differences and automatically resolves these problems or displays them for manual resolution.

By facilitating parallel rather than serial methods of software development, the NSE promotes network-based development. Using the network-based configuration and version control system, users easily control and manage complex distributed software projects. From within the NSE, developers can use all standard Sun software development tools, including the standard Sun compilers, linkers, and debuggers, and all UNIX editors and utilities. With a consistent and standard mechanism for software tool integration, the NSE encourages users to adopt emerging technologies as they become available.

The NSE is a generic platform for the integration of computer-aided software engineering (CASE) tools. A programmatic interface to all NSE functions assures a consistent and standard mechanism for software tool integration and allows users to configure tool sets into a single development environment. By using the standard NSE extension facilities, users can apply the NSE to non-software development projects.

Users can configure and extend the NSE open architecture to meet specific software development needs by using NSE capacity for defining new types. Rather than limiting development objects to a predefined set, the NSE allows developers to define unique application-specific objects. Users cannot tell which objects are built-in and which have been added. This feature offers users a single view of development objects regardless of their origins. In addition to supporting extension via add-in objects and tools, the NSE also provides a standard way to link objects, or parts of objects, together. Thus, new objects can be integrated both with the NSE and with each other.

For increased accessibility, the NSE includes two user interfaces. Powerful window- and mouse-based graphic interfaces fully utilize the high-resolution, bit-mapped graphics available on Sun systems and the alternate command line interface.

For more information on NSE and its functions, the following documents are recommended:

- *Introduction to the NSE*
- *Network Software Environment: Reference Manual*
- *Network Software Environment: Installation and Administration*

4.3 SunTrac Project Management System

SunTrac puts control into the hands of project managers

SunTrac blends sophisticated analysis with networked computing technology to put control into the hands of project managers. By incorporating Sun's bit-mapped graphics displays, multitasking window system, and mouse- and menu-driven user interface, SunTrac quickly responds to project changes and communicates results to project participants.

Modern management strategies produce dense, highly interconnected project networks with many near-critical paths. Although these strategies maximize resource use and aid just-in-time materials procurement, they are vulnerable to disruption by unforeseen delays. SunTrac's Risk Analysis feature provides a way to plan for disruptions and recover from them.

SunTrac is an easy-to-use risk analysis tool. It analyzes the interplay of all activities, calculates how their effects on each other will affect cost, schedule, etc., then expresses the results in probabilistic terms to accurately identify risk to the schedule. The manager can then run what-if analyses at different levels of certainty to see additional cost and schedule effects.

SunTrac features:

- Traditional Gantt, PERT, and Critical Path Method diagramming and analysis in a powerful graphic/interactive environment
- Graphic display and presentation-quality output
- A unique risk assessment algorithm
- Automatic generation of cost-optimized overtime plans
- Graphically interactive resource leveling
- ASCII file structure
- Hierarchical framework

SunTrac benefits:

- Delivers rapid what-if and realtime analysis capability
- Provides instant communication of complex project relationships
- Assesses the probability of success at any management level, at any time during the project's lifecycle
- Accelerates the schedule realistically
- Pinpoints scheduling options and resource use by activity
- Easily imports and exports data to other applications or hardware
- Encourages involvement by the entire staff and leverages the power of the network

Management quickly defines a project's goals and milestones, then assigns principal areas of responsibility to subnetworks in the top-level diagram. Staff members then expand the subnetworks with bottom-up estimates of task duration and staffing levels. The expanded subnetworks form a hierarchy of plans for the project, any of which is instantly available over Sun's Network File System (NFS). The hierarchy offers a range of detail from individual tasks to summary top-level overviews.

File formats are published in the *SunTrac Reference Guide*.

For additional information on SunTrac, please refer to the following documents:

- *SunTrac User's Guide*
- *SunTrac Reference Guide*
- *SunTrac Tutorial*
- *SunTrac Installation Guide*