# SDK Components 2

The SDK CD-ROM disk contains the following:

- On-line demonstrations
- Software
- Documentation

The software and documentation are called the *components* of the SDK. This chapter describes each SDK component and tells you about the documentation associated with that component. If a component has features that are new in the Solaris 2.5 release, they are listed. At the end of this chapter, documentation that supports software development, but is not closely associated with a software component of the SDK, is listed and described.

## On-Line Demonstration

The on-line SDK demonstration is a series of on-line technology demonstrations and documentation that feature the SDK components. This demonstration includes:

- Technology demonstrations
- Source code for each demonstration
- Technical white papers
- Technical newsletters
- Referrals to helpful third-party books

# 2

Preview this on-line demonstration for a better understanding of features and functions offered by the SDK components. A Motif graphical user interface (GUI) provides you with easy access to each demo, white paper, and newsletter.

## *To Display the Demonstration*

1. **Insert the caddy containing the SDK CD-ROM disc into the drive.**

---

**Note –** If you have the volume management software enabled (the default with the Solaris 2.5 release), inserting the CD-ROM disc mounts it automatically. If you have disabled volume management, mount the CD-ROM disc using the `mount(1M)` command.

---

**Note –** If you are installing from a remote CD drive, refer to the *Solaris 2.5 Software Developer Kit Installation Guide* for instructions on installing the SDK from a remote CD drive.

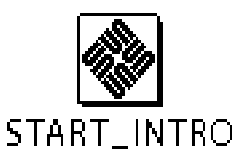
---

**Note –** To run the SDK on-line demo, your system must support Solaris audio output (that is, `/dev/audio` must point to a valid audio device).

---

File Manager opens, displaying the contents of the CD-ROM disc, including the START_INTRO icon.

## *From File Manager*




2. **Click twice on the START_INTRO icon**
   A dialog box opens and asks if you want to run the demonstration in a shell tool or without one.

3. **Choose "No Shell Tool"**
   In a few minutes, you will see a window entitled *Welcome to the Solaris 2.5 Software Developers's Kit*. Wait a few more minutes and the window displaying the SDK-demonstration Categories and Contents appears. See Figure 2-1.

**Note –** If you get the error message `sh: START_INTRO not found`, you don't have "." in your path. You must run the demonstration from the command line. See the section "From the Command Line" on page 8".

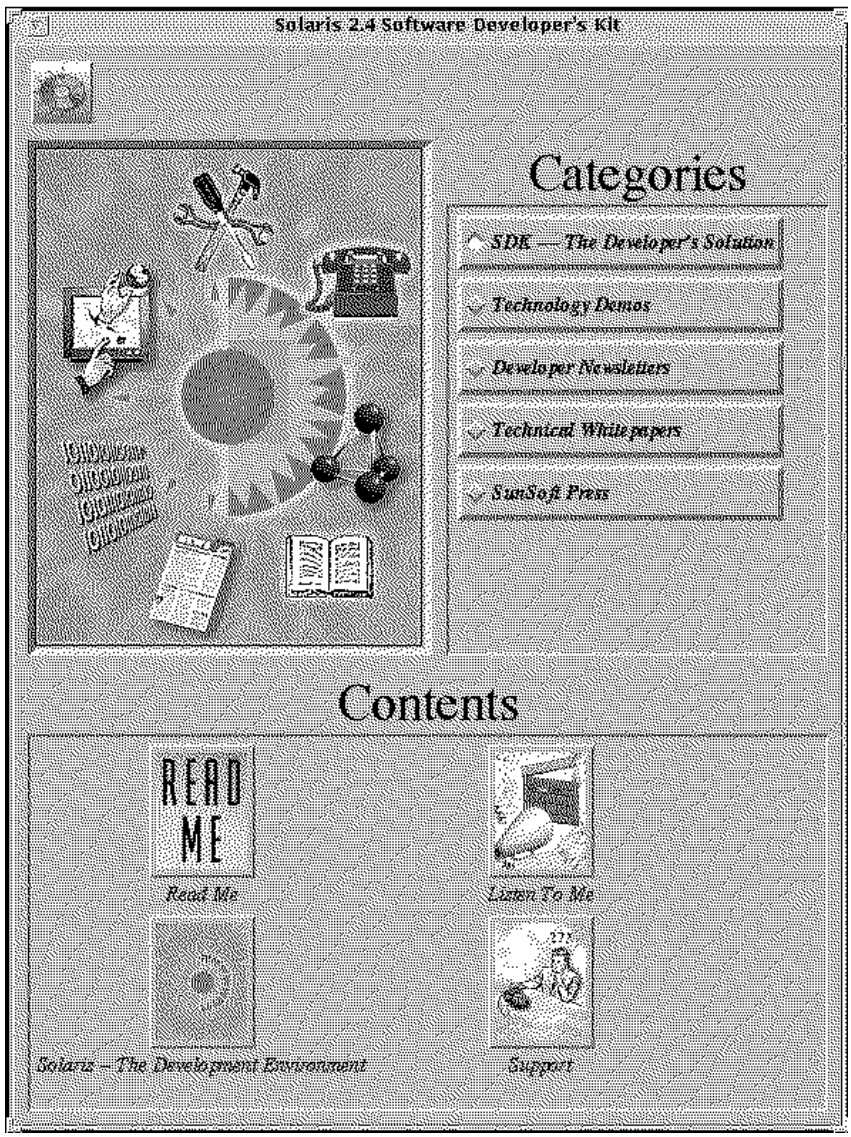



*Figure 2-1* Categories and Contents Window

## *From the Command Line*

**1. From the command line enter:**

```
% /cdrom/cdrom0/START_INTRO
```

In a few minutes, you will see a window entitled *Welcome to the Solaris 2.5 Software Developer's Kit*. Wait a few more minutes and the window displaying the SDK-demonstration *Categories* and *Contents* appears. See Figure 2-1.

Once the Categories and Contents window appears you can choose to look at:

- The SDK overview, *SDK—The Developer's Solution*
- Technology demos
- Developer newsletters
- Technical white papers
- SunSoft Press

As you choose a category, the *Contents* section of the window changes to reflect the category you have chosen. If you choose the *Technology Demos* category, you will always see the following icons within a component selection:

- The executable file that starts the demo.
- The *Listen To Me* icon—An audio description of the technology.
- The *Read Me* icon—A description of the system requirements and an explanation of the demonstration.
- The *What's Next* icon—Step by step instructions of what to do next. For example, install software, read certain white papers, read third-party books, location of source code for the demonstration.

You will also see, at the bottom of the screen, a written overview of a component.

---

**Note –** The SDK on-line technology demonstrations and corresponding source code are provided as a technology demonstration only. SunSoft™, Inc. does not support the demonstrations nor the source code. SunSoft, Inc. makes no commitment to enhance the demo or fix any bugs that may exist in the code.

---

## Internationalization

The Solaris development environment is fully internationalized, enabling you to build applications that conform to *level-4 internationalization*. This means that this Solaris release:

- Is 8-bit clean
- Adheres to standard formats and collation in displaying:
  - Money
  - Numbers
  - Date
  - Time of day
- Supports messaging in native locale
- Supports multibyte characters, allowing for support of Asian character sets

Using the internationalized features of the Solaris development environment, you can create applications that can be localized for any language.

### Documentation

See the *Developer's Guide to Internationalization* for more information on Solaris internationalization. This guide describes both internationalization and localization of applications, formats, and conventions in different countries, internationalization facilities and their use, and various special considerations.

## Multithreaded Programming

The operating system kernel is multithreaded, allowing for asymmetric, multiple processors to execute the kernel at the same time. This allows for application concurrence, or the execution of applications written as independent computations. The multithreaded kernel is on the Solaris CD-ROM disc.

### Documentation

For further information, see the *Multithreaded Programming Guide*, which introduces multithread programming concepts and describes basic use of threads, synchronization functions, and the Solaris two-level model.

# 2

## STREAMS Framework

STREAMS, a framework for character I/O, is implemented throughout the operating system. The STREAMS framework provides you with:

- Integral functions
- A set of utility routines
- Facilities that expedite software design and implementation

### Documentation

For further information on the STREAMS framework, see the *STREAMS Programming Guide*, which provides information on the use of the STREAMS mechanism at user and kernel levels.

## Network Applications

In a Solaris development environment, you can write network applications without coding the details of specific protocols and address formats. This is network transport independence. The Solaris programming environment provides these capabilities through the ONC+ facilities. ONC+ is a family of network protocols for connecting large heterogeneous networks.

Remote procedure call (RPC), which is included in ONC+, enables an application to execute procedures on remote systems over a network. The transport layer interface (TLI) makes a protocol-enhanced RPC, enabling RPC programs to run across multiple network transports. External data representation (XDR) in ONC+ allows applications to exchange data across heterogeneous hardware systems. The `rpcgen` tool for RPC is included as a standard part of the Solaris programming environment. You can use the `rpcgen` tool to generate templates for client programs and server routines in a high-level, C-like language called RPC language. The code generator creates the necessary RPC code to be included in the client and server pieces of the application.

### Federated Naming Service

The federated naming service (FNS) provides support for flexible composition of different, autonomous naming systems into a single service, accessible using a single, simple naming system interface. FNS facilitates coherent naming by

encouraging the use of common naming policies. These interfaces and policies are used and shared by systems and applications, such as the file service and print service, to provide a coherent view of the global and enterprise-wide namespaces in the Solaris environment.

FNS exports the X/Open Federated Naming (XFN) programming interface and policies, an open standard actively supported by other vendors. This enhances the portability of applications that use FNS.

In the Solaris 2.5 release, FNS consists of XFN libraries, administration tools, and a set of XFN-conformance naming systems implemented using NIS+ for naming organizations, users, hosts, sites, services, and files within the enterprise.

#### *Documentation*

Refer to the *Federated Naming Service Guide* for details on installing and administrating the FNS environment.

### *Federated Domain Name System*

Global naming services such as Domain Name System (DNS) have world-wide scope. FNS provides name resolution for hosts and domains in the Internet Domain Name System. Names can be constructed for any enterprise that is accessible on the Internet, as well as objects exported by these enterprises.

### *Federated X.500*

X.500 is another global naming service. FNS provides name resolution for entities in the X.500 namespace. Names can be constructed for any enterprise that is accessible through X.500, as well as objects exported by these enterprises.

### *FNS-based File Naming*

FNS-based file naming integrates FNS naming into the Solaris file service. FNS-based file naming enables files to be named relative to users, hosts, sites, and organizations, using the FNS policies shared with other non-file applications. FNS-based file naming gives clients a common view of the global and

enterprise-wide file namespaces. Solaris applications that access the file system will, without modification, have access to the file namespaces supported by FNS.

### *FNS-based Printer Naming*

FNS-based printer naming provides the basic naming support for the unbundled new printing client product. FNS-based printer naming enables printers to be named relative to users, hosts, sites, and organizations, using the FNS policies shared with other non-printing-related applications. FNS-based printer naming gives clients a common view of the global and enterprise-wide printer namespaces and allows centralized administration of the printer namespaces.

## *Multithreaded-Safe Server*

The *multithreaded-safe server* remote procedure call (RPC) interface was new in the Solaris 2.4 release. The *multithreaded-safe client* RPC interface was new in the Solaris 2.3 release. You may invoke these interfaces to put a server into one of two multithreaded modes: *auto mode* or *user mode*. The auto mode is easy to use and powerful. The user mode requires that you know more about multithreaded interfaces and issues, but offers more flexibility.

### *Documentation*

These RPC interfaces are described in the *Transport Interfaces Programming Guide*.

## *Additional Network Applications Documentation*

For further information, see the following manuals:

- *ONC+ Developer's Guide*
- *Transport Interfaces Programming Guide*
- *Federated Naming Service Guide*

These manuals describe the use of basic, intermediate, and high-level facilities to:

- Develop distributed applications

- Convert existing applications from single-computer to distributed-computer
- Maintain distributed applications in the Solaris 2.x environment.

## ToolTalk Messaging Services

The ToolTalk service included in the Solaris development environment is a network-spanning, interprocess messaging system. Applications can create, send, and receive messages by calling ToolTalk™ functions from the ToolTalk application-programming interface (API) library. The ToolTalk service enables independent applications to communicate with other applications without having direct knowledge of each other. The ToolTalk service allows one application to effectively be replaced with another application as long as both applications support the same ToolTalk interface.

### Documentation

For further information on the ToolTalk service, see:

- *ToolTalk User's Guide*—This guide describes ToolTalk. It discusses ToolTalk functionality, including modifying applications to use ToolTalk. It describes ToolTalk messaging and message sets. Use this guide if you are creating or maintaining ToolTalk applications, or if you are a system administrator supporting ToolTalk developers.
- *ToolTalk Reference Manual*—This manual describes the API components, commands, and error messages.
- *ToolTalk White Paper*—This paper provides a brief and high-level discussion of ToolTalk features and benefits. Read this paper if you want a quick overview of ToolTalk.
- *Introduction to the ToolTalk Service White Paper*—This paper gives an introduction to ToolTalk. It discusses how ToolTalk can fit into a system and a product strategy. This paper is for non-developers who want to understand what ToolTalk is and how it can relate to their products.
- *The ToolTalk Service White Paper*—This papers gives a technical introduction to ToolTalk, models, services, and terminology. It contains an overview of all basic ToolTalk capabilities, what they do, and how you can use them. Read this paper if you want to evaluate or use ToolTalk.

- *Designing and Writing a ToolTalk Procedural Protocol*—This paper describes the general principles for designing, writing, and implementing ToolTalk protocols.
- *The ToolTalk Desktop Services Message Set White Paper*—This paper describes a basic set of ToolTalk messages that are designed to facilitate control and cooperation among clients on a desktop (for example, launching, halting, control, or passing output from one client to another).
- *The ToolTalk Document and Media Exchange Message Set White Paper*—This paper describes a set of ToolTalk messages designed to facilitate the exchange of multimedia data (for example, graphics, sound, video) among cooperating applications.
- *A Simple Demonstration of How the ToolTalk Service Works*—This paper explains how to design and build applications that interoperate with other applications that are using the ToolTalk service.

## Object Products

The NEO product family represents a new approach to building enterprise-wide software systems. Because object-based programs support reuse, you can construct complex applications easier than before. If you are developing networked objects, NEO supports the creation of applications that are substantially faster to construct and easier to evolve.

### Documentation

For further information on NEO products, see:

- NEO white papers—These papers provide an overview of NEO technology, including the operating environment, developer and management tools, and available connectivity and migration products.
- *The Messaging Object Service*—This paper describes the messaging-object-service for NEO objects and how you can use this service in your object oriented applications. It also describes how applications can send, specify, and receive other applications.

# Solaris VISUAL

The Solaris VISUAL environment includes several graphics and multimedia software *foundation libraries*. Foundation libraries are the lowest-level device-independent layer of the Solaris software. This level of interface is designed to support a wide variety of common functions. You can build higher-level libraries on top of the foundation libraries, or the foundation libraries can be used directly by a software application. These foundation libraries enable you to create applications that incorporate 2-D and 3-D graphics, imaging, and digital video. The libraries are the XGL graphics library (which serves as a foundation library for Solaris PEX) and the XIL imaging library.

The Solaris VISUAL environment also includes the OpenWindows X Server and libraries (X11/Xlib) and the Display PostScript (DPS) libraries.

## Documentation

For further information on the Solaris VISUAL environment, see the *Solaris Visual White Paper*, which gives an overview and philosophy of the Solaris VISUAL.

## Solaris XGL 3.2 Graphics Library

The Solaris XGL Graphics Library is a graphics interface, providing foundation-level geometry graphics support for Solaris-based applications. The XGL graphics library is both an API and a geometry foundation library that provides:

- A set of low-level graphics primitive operations for 2-D and 3-D geometric applications.
- High performance rendering capabilities for a wide variety of graphics operations across numerous graphics hardware platforms.
- An API that presents an immediate-mode graphics programming model.
- Support for graphics-based applications, such as CAD programs.
- A foundation library for higher-level graphics APIs, such as PEXlib, PHIGS, and GKS.

The XGL runtime library is delivered on the Solaris 2.5 CD-ROM disc. The XGL developer libraries and documentation are delivered on the SDK CD-ROM disc.

### *Documentation*

For further information on the XGL graphics library, see the following documents:

- *XGL Accelerator Guide for Reference Frame Buffers*—This guide provides performance hints for application programs that use the XGL library and SunSoft reference hardware platforms.
- *XGL Programmer's Guide*—This guide provides information and example programs that you can use if you are writing applications using the XGL graphics library.
- *XGL Reference Manual*—This manual includes the manual pages that describe the syntax for using the operators (functions) and attributes in the XGL graphics library.
- *The XGL White Paper*—This paper describes the purpose, structure, and features of functions in the XGL graphics library.

### *New Features*

The following is a list of new XGL features with this release:

- Raster text
- Environment texture mapping
- Vertex-level texture mapping
- 4-component texture mapping
- DGA transparent overlay
- Triangle list Gcache

## *Solaris PEX 3.0*

Solaris PEX 3.0 RTE consists of a PEX client-side runtime library (PEXlib) and a PEX server-extension shared-object file (Solaris PEX server). PEXlib is an application programmer interface (API) to PEX protocols as Xlib is to core X protocols. PEXlib provides application portability across hardware platforms and enables 3-D graphics rendering on local and remote displays. The OpenWindows Version 3.5 X11 server automatically loads the PEX shared-object extension when PEX requests are sent from a PEX client.

The PEXlib implementation for Solaris 2.5 is based on the *PEXlib - Specification and C Language Binding: Version 5.1*, available from the Massachusetts Institute of Technology (MIT). The PEX server extension is based on the *PEX Protocol Specification, Version 5.1*.

## *Installation Hint*

The Solaris PEX 3.0 RTE is included in the Solaris 2.5 release for your convenience and should be installed if you are planning to:

- Run PEXlib-based graphics client applications
- Display any 3-D graphics from client applications that render on remote displays through the PEX protocol

## *Supported Devices and Features*

Solaris PEX 3.0 RTE supports all the devices that are supported in XGL 3.2 through Direct Graphics Access (DGA) and Xlib.

---

**Note –** See the *Solaris 2.5 Software Developer Kit Installation Guide* for special instructions on how to install the PEX software packages.

---

## *Documentation*

For further information on PEX, see the following documentation.

- Available with the SDK:
  - *Solaris PEXlib Reference Manual*—This manual contains manual pages for standard PEXlib, Sun extension, and CGE extension functions.
  - *Solaris PEX Implementation Specification*—This specification describes PEX software implementation and limitations.
  - *CGE PEX 5.1 Portability Guide*—This guide assists you in creating portable and interoperable PEXlib applications across platforms supporting the CGE.
- Available through a computer book store:

- *PEXlib Programming Manual*, O'Reilly and Associates, Inc.—This manual describes PEXlib, the programming library for 3-D graphics, and how to use it. It also recommends how to write PEXlib programs for better interoperability across any vendor's PEX implementation.
- *PEXlib Reference Manual*, O'Reilly and Associates, Inc.—This manual contains manual pages for each PEXlib routine.

## *Solaris XIL 1.2 Imaging Library*

The XIL 1.2 Imaging Library is a foundation library that provides image processing and digital-video applications. The ISV interface enables you to access the functions in the library as C subroutines. The XIL imaging library supports:

- Image processing
- Image compression
- A wide variety of general digital video applications
- Higher level APIs

All users who are running the Solaris operating system can run XIL applications because the XIL runtime environment is included with the Solaris software.

### *Documentation*

For further information on the XIL imaging libraries, see:

- *XIL Programmer's Guide*—This guide describes, in detail, how to use the XIL imaging library to create imaging or video applications. It includes discussion of input, output, and error handling. It also describes image-processing functions and covers using the XIL imaging library to compress and decompress sequences of digital images.
- *XIL Reference Manual*—This manual contains all the manual pages available with the XIL imaging library.
- *The XIL White Paper*—This paper describes the purpose, structure, and features of functions in the XIL imaging library.

## Solaris X Server and Libraries

OpenWindows, the X-based window system environment, is a standard part of the Solaris development environment. The OpenWindows environment is network based, offering the X protocol interface. An application that you write in this environment can take advantage of multiple machines distributed over the network. The X and Display PostScript (DPS) libraries and the Solaris X server are on the Solaris CD-ROM disc.

### Documentation

For more information on these libraries and the Solaris X server, see the following manuals:

- *Solaris X Window System Developer's Guide*—This guide instructs you in writing applications based on X11 and Display PostScript, that run in the OpenWindows environment.
- *OpenWindows Desktop Reference Manual*—This manual contains the manual pages for the OpenWindows DeskSet™ tools.
- *Solaris X Window System Reference Manual*—This manual contains the manual pages for the Solaris X server and libraries.

### New Features

The following new features are provided in this release of the Solaris X server:

- Shared memory transport extension
- MT-safe DGA
- DPS toolkit for Motif
- NumLock key support
- DPS composite operators

See the *Solaris X Window System Developer's Guide* for more information.

## Kodak Color Management System 1.0

The Kodak Color Management System (KCMS) software product is a color management solution that ensures color consistency from input devices to output devices. It is technology licensed from Eastman Kodak that was

developed in conjunction with SunSoft. The KCMS product is fully integrated with SunOS—it is bundled with this Solaris release—and has SDK and DDK components.

Consistent color is difficult to achieve because of the many ways devices produce color even between the same model of devices. For example, the colors you choose on your monitor are based on that particular manufacturer's monitor set up; they are in a device-dependent color space. If you display these same colors on a different monitor, they will appear differently.

The KCMS product provides a C-based API that hides the color science required to provide color consistency. It provides you with a paradigm for connecting one device to another. Devices are represented by profiles that encapsulate the color properties of a device or color space and provide the data for the KCMS framework.

Many companies involved with color management have established a consortium called the ICC. This consortium defined an ICC profile with a standard file format for storing device color characteristics. The KCMS product supports and provides default ICC profiles. In addition, you can share ICC profiles between different operating system vendors and color management vendors. For more information on the ICC profile format, see the ICC specification located on-line in `/opt/SUNWsdk/kcms/doc/icc.ps`.

The KCMS API consists of a set of C functions, header files, and a shared library and dynamically-loaded modules. Use the KCMS API when writing an application that:

- Uses color data
- Prints
- Is an imaging tool
- Uses PhotoCD

The SDK provides KCMS header files and sample code to incorporate color management into your applications. The SDK also includes a sample implementation of a monitor calibration device that interfaces with the calibration program, `kcms_calibrate(1)` to gather monitor luminance values for the calibration of monitor profiles. This sample is implemented with an X Window System interface and a device interface through RS232 ports to a colorimeter. The source code files for this sample implementation are located on-line in `/opt/SUNWsdk/kcms/src`. The documentation describing this sample interface is on-line in

`/opt/SUNWsdk/kcms/doc/calibrator_design.ps`. The documentation explaining how to use Calibrator Tool provided on the Solaris 2.5 CD-ROM disc is on-line in `/opt/SUNWsdk/kcms/doc/calibrator_guide.ps`.

The KCMS runtime library is delivered on the Solaris 2.5 CD-ROM disc. The KCMS developer libraries and documentation are delivered on the SDK CD-ROM disc.

### *Documentation*

For further information on the KCMS library, see the following document:

- *KCMS Application Developer's Guide*—This guide provides information and example programs that you can use if you are writing applications using the KCMS library.

## *Windows Graphical User Interface Toolkits*

Three window graphical user interface (GUI) toolkits available for application development:

- Motif
- XView
- OLIT

The Motif toolkit supports the Motif graphical user interface; XView and OLIT support the OPEN LOOK graphical user interface. All three toolkits are delivered as a standard part of the Solaris release and provide the routines and libraries for designing user interfaces and handling events such as mouse pointing and user input.

The Motif toolkit is the SunSoft strategic toolkit for the future. If you are planning to develop new applications, SunSoft strongly encourages you to use the Motif toolkit. If you have existing applications based on XView or OLIT you can continue to do application enhancement or maintenance using the XView or OLIT toolkits.

# 2

## *Motif Toolkit*

The Motif toolkit gives you a head start in creating applications for the *Common Desktop Environment* (CDE). It includes a comprehensive set of tools that include the:

- Motif toolkit (libXm) & Motif Resource Manager (libMrm) libraries
- Motif window manager
- User interface language (UIL) compiler

The Motif toolkit is based on the industry standard X Window System and X Toolkit Intrinsics from the X Consortium. It works with X11R5 libraries, making it portable across a range of hardware platforms. The Motif toolkit also offers a set of drag-and-drop operations, enabling you to create graphical user models for data selection and transfer. The UIL compiler enables rapid user-interface design and prototyping without requiring massive amounts of code, significantly reducing application development time.

The Motif runtime libraries and header files are on the Solaris 2.5 CD-ROM disc in the `SUNWmfrun` package. The `SUNWmfrun` package is in the All or Developer cluster of the Solaris 2.5 release.

### *Documentation*

For further information on the Motif toolkit, see the following documents:

- *OSF Application Environment Specification (AES) User Environment Volume*, Revision C, PTR Prentice Hall—This specification describes how to use the Open Software Foundation (OSF) and Motif GUI. It describes how to manipulate windows on-screen, use applications written for the Motif environment, and customize the Motif desktop environment.
- *OSF/Motif Programmer's Guide*, Release 1.2, PTR Prentice Hall—This guide describes how to use the OSF/Motif API to create Motif applications. It gives an overview of the Motif widget-set architecture, explains Motif toolkit features, and presents a model and examples for constructing Motif applications.
- *OSF/Motif Programmer's Reference*, Release 1.2, PTR Prentice Hall—This manual describes the API for the OSF/Motif GUI. It is a reference manual for OSF/Motif commands and functions (it describes all toolkit, window manager, and user interface language commands and functions).

- *OSF/Motif 1.2 Style Guide*, PTR Prentice Hall—This guide describes the behavior specifications of the OSF/Motif GUI. It guides application developers, widget developers, user interface system developers, and window manager developers in the design and implementation of new products consistent with the OSF/Motif user interface.
- *Solaris CDE Motif Transition Guide*—This guide discusses issues of concern to Sun Motif developers. For example, how to run existing OPEN LOOK and Motif applications on the OpenWindows 3.5 and Solaris CDE 1.0 desktops; and Porting OPEN LOOK and Motif applications to the Solaris CDE environment.
- *Transport Interfaces Programming Guide*—This guide describes how to migrate from OPEN LOOK to Motif with the Motif 1.2.3 toolkit.

## *XView Toolkit*

The XView toolkit is a user-interface toolkit based on the X11 Windows System. It supplies you with a set of prebuilt OPEN LOOK objects such as buttons and menus. You can use these objects to build applications that are compliant with OPEN LOOK.

### *Documentation*

For further information on the Motif toolkit, see the *XView Developer's Notes*, which include XView developer information not present in the O'Reilly XView documentation set. These notes contain two parts:

- Motif interoperability and XView panel issues
- Internationalizing XView applications

## *OPEN LOOK Intrinsics Toolkit*

The OPEN LOOK Intrinsics Toolkit (OLIT) is a widget set and library based on the X Window System that you can use to create applications supporting the OPEN LOOK graphical interface. Widgets are user interface objects such as:

- Buttons
- Scrollbars
- Control areas
- Text edit areas

- Drawing areas

When you create and manipulate widgets with the OPEN LOOK Intrinsics Toolkit, you can create an OPEN LOOK user interface for your application.

### *Documentation*

See the following documents for more information on OLIT:

- *OLIT QuickStart Programmer's Guide*—This guide presents fundamental OLIT programming concepts and provides a foundation for more advanced OLIT topics.
- *OLIT Reference Manual*—This manual provides descriptions of the OLIT widget set and related convenience functions that you can use with the widget set. It guides you in developing internationalized OLIT applications using these widgets and functions.

## *OpenWindows 3.0.1 Developer's Guide*

The OpenWindows Developer's Guide (Devguide) is a GUI builder for developing OPEN LOOK applications. It is designed to make your job easier. Devguide is a programming tool that lets you create and try user interfaces without writing any code. Because you can create and modify an interface easily, you can concentrate on designing and testing interfaces and enhancing functionality, rather than on coding.

Once you finish with interface design, you can save significant time and reduce cost by using one of the included code generators, `gxv` or `golit`, to generate XView or OLIT application code.

Most of the principles common to the OpenWindows environment are present in Devguide. You can select, drag, and drop objects using a mouse to quickly create and test an interface design before any code is created. Devguide enables you to create applications that integrate with the Solaris DeskSet environment, not only in look and feel, but also in interoperability. This capability provides you with a consistent and easy-to-use desktop.

Devguide also enables you to create applications that can be easily localized for the international market.

### *Documentation*

For more information on Devguide see the following documents:

- *OpenWindows Developer's Guide: User's Guide*—This guide provides a comprehensive description of Devguide.
- *OpenWindows Developer's Guide: Motif Conversion Utilities Guide*—This guide describes `GMF` and `GUIL` utilities used with Devguide user interface GIL files to produce Motif C code or Motif UIL files.
- *OpenWindows Developer's Guide: XView Code Generator Programmer's Guide*—This guide describes `gxv` and `gxv++`, the XView toolkit code generators used with Devguide user interface GIL files to produce XView C code.
- *OpenWindows Developer's Guide: OLIT Code Generator Programmer's Guide*—This guide describes `golit`, the OLIT code generator used with Devguide user interface GIL files to produce OLIT C code.

### *Devguide Motif Utilities*

Devguide Motif utilities enable you to migrate Devguide-generated OPEN LOOK applications to Motif. The code generators, `gmf` and `guil`, take the Devguide user interface GIL files and output Motif C and Motif UIL source code, respectively. You can continue to add C code to the Motif or UIL code to complete an application. The generated UIL Motif code can also be fed back into a third party Motif GUI builder for further development and testing.

#### ***Documentation***

For further information on Devguide Motif utilities, see the *Transport Interfaces Programming Guide*, which describes how to migrate from OPEN LOOK to Motif applications with the Motif 1.2.3 toolkit.

## *Supplemental On-Line Documentation*

In addition to the documents described in the component sections of this chapter, the following on-line documentation is also provided with the SDK. This documentation is available in three on-line documentation sets called:

- *Solaris 2.5 Software Developer AnswerBook*
- *Solaris 2.5 Reference Manual AnswerBook*

- *Solaris 2.5 Supplemental Developer AnswerBook*

For a complete list of SDK documentation, see chapter 3, "Documentation."

These documents are in the *Solaris 2.5 Software Developer AnswerBook* on-line documentation set:

- *System Interfaces Guide*—This guide describes the operating-system services. It also provides information for experienced programmers and system administrators writing or administering real-time applications.
- *Standards Conformance Guide*—This manual describes how the Solaris programming environment and the operating system comply to such industry standards as the:
  - *Application Binary Interface, X/Open™*
  - *IEEE Standard 1003.1 (Posix.1)*
  - *SPARC Compliance Definition*

  If you adhere to the guidelines in this manual, you can write applications that are consistent across conforming systems.
- *Programming Utilities Guide*—This guide describes the following built-in operating system programming tools: `m4`, `make`, `sccs`, `lex`, `prof`, `yacc`, and System V `make`.
- *x86 Assembly Language Reference Manual*—This manual describes the x86 Assembler. It includes the Assembler syntax, executable and linking format, and instruction-set mapping.
- *SPARC Assembly Language Reference Manual*—This manual describes the SPARC Assembler. It includes the Assembler syntax, executable and linking format, conversion of files, and instruction-set mapping.
- *Linker and Libraries Guide*—This manual describes the operations of the operating system linker and libraries, summarizes their mechanisms, and describes how to use them to build executable programs and libraries.
- *Source Compatibility Guide*—This guide describes how the SunOS/BSD Source Compatibility Package supports the SunOS 4.x commands, library routines, and files in the SunOS 5.4 environment.
- *Desktop Integration Guide*—This guide instructs ISVs who wish to integrate their applications with other applications and tools that run in the OpenWindows environment.

The following documents are provided in the *Solaris 2.5 Reference Manual AnswerBook* on-line documentation set:

- *SunOS Reference Manual*—This manual contains the SunOS 5.4 manual pages. These manual pages are organized by section as follows:
  - *man Pages(1): User Commands*
  - *man Pages(1M): System Administration Commands*
  - *man Pages(2): System Calls*
  - *man Pages(3): Library Routines*
  - *man Pages(4): File Formats*
  - *man Pages(5): Headers, Tables and Macros*
  - *man Pages(6): Demos*
  - *man Pages(7): Device & Network Interfaces*
  - *man Pages(9): DDI and DKI Overview*
  - *man Pages(9E): DDI and DKI Driver Entry Points*
  - *man Pages(9F): DDI and DKI Kernel Functions*
  - *man Pages(9S): DDI and DKI Data Structures*

The following document is provided in the *Solaris 2.5 Supplemental Developer AnswerBook* on-line documentation set:

- *Application Packaging Developer's Guide*—This guide describes how to create an installable package for application software and gives examples of package creation. It also describes how to create a CD-ROM disc that contains a package.

# 2