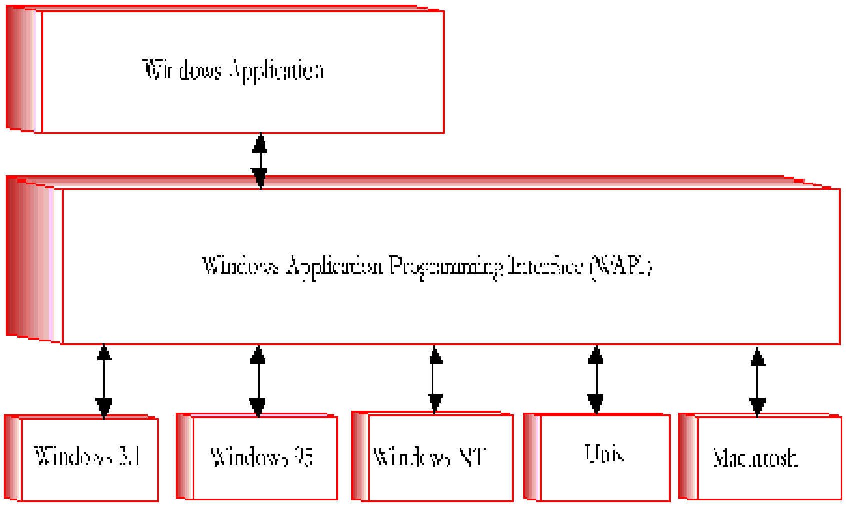
The Willows Toolkit Technical White Paper

This document addresses the 16-bit version of The Willows Toolkit formerly known as TWIN APIW. This white paper will soon (December 1996) be updated to reflect the new product architecture and capabilities.

The Willows Cross-Platform Windows Library (The Willows Toolkit) is a Windows implementation that allows windows application developers to migrate their applications to multiple platforms, including UNIX and Macintosh, using a single source base, which preserves native platform performance.

The Willows Cross-Platform Windows Library provides a platform-independent interface layer between Windows compatible applications and the underlying target environment. From the application perspective, this interface layer functionally appears to be a Windows environment with all the features and functions that the application expects. The developers are saved from having to master new environments when moving their application to a new platform, significantly reducing time to market.

The Willows Cross-Platform Windows Library is a natural extension of the use of the Windows API on Windows NT as a growth path from the traditional PC platform. It is the recognition of this de facto standard that is driving the international development community to standardize the Windows API in what has become the APIW Standard.



The Willows Application Programming Interface (WAPI) is an implementation of the Windows API that provides the windows environment for a windows application on the Macintosh and Unix.

The Willows Application Programming Interface is made up of three major components:

- Willows Binary Interface
- Willows Library API
- Willows Drivers

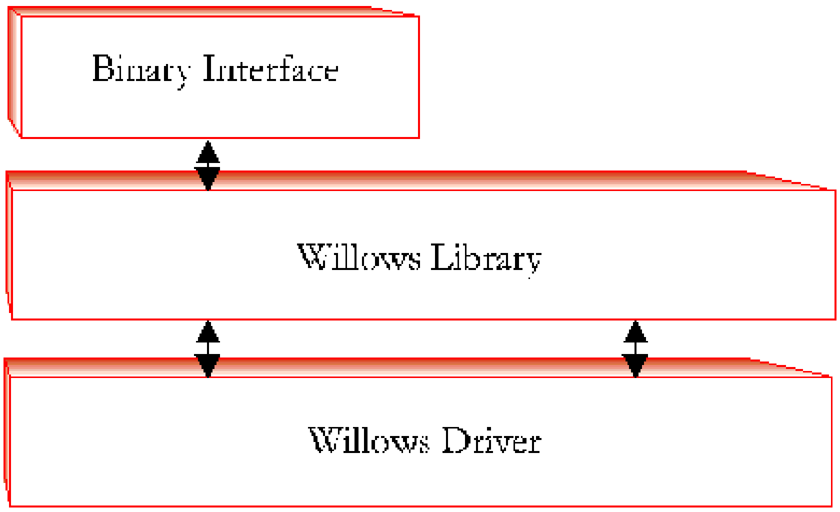
The Windows compatible application interacts with the Willows Library in the same manner as the application works in the

Windows environment, calling API functions, receiving messages, loading resources and even launching other Windows applications.

Existing Windows applications and DLLs interact with the Willows Binary Interface to access the native Willows API on non-Intel platforms. This layer translates all Windows API requests to the native Willows Library API to allow a non-native application to get native performance.

In the same manner, the Willows Library interacts with the driver layer, making requests for graphical, window, or system operations and receiving responses or asynchronous messages.

The Willows Application Programming Interface is capable of recognizing binary Intel objects such as third-party dynamically-linked libraries (DLL), Visual Basic controls (VBX), drivers (DRV) and custom controls. This unique facility allows the inter-mixing of source and binary modules within an application. This feature allows developers to bring their applications to market faster, without having to wait for the third-party market to port the libraries. Resource files, bitmaps and icons can continue to be used without change. Applications can continue to load resources from existing DLLs, without modification.



The binary interface consists of four sub-systems that provide the necessary functionality and environment to load and execute existing window binary applications, drivers and dynamic link libraries. The four sub-sections are:

- Virtual Machine
-
- DOS Emulator
-
- 16/32 Thunking Layer
-
- 16/32 Loader
-

The Virtual Machine provides the hardware abstraction layer that simulates an Intel Microprocessor running in protected mode. This layer manages code and data segments, Intel registers, interrupt control, instruction and I/O emulation. All native addresses are mapped to selector/offset addresses, including data addresses and window API function addresses.

The DOS Emulator emulates the DOS interrupt and functional interface. All DOS functions supported by the Windows API are implemented in this layer, including process management and file I/O operations. Access to serial and parallel ports is done by mapping them to native devices or output filters. Floppy disk support is included, which allows files on DOS floppy disks to be accessed using the standard DOS for files on floppy disks. DOS hard drive support is implemented through the use of global and user-defined aliases, which map hard disk filenames to the native file system.

The 16/32 Thunking Layer is responsible for handling calls between the 16-bit windows code and the native Willows library. The 16-bit code runs in protected mode, using selector/offsets for addresses, and a 16-bit stack. The 16/32 Thunking Layer handles calls from the 16-bit space into the native space as well as calls from the native space to the 16-bit space. Each call is identified by a 'method' that handles a specific type of call. A given method can range from something as simple as calling the library and returning the result in the appropriate registers, to something complex that involves converting addresses and data structures on both the call and return from the call. Thunks allow for hybrid applications that are a mixture of both 16- and 32-bit code and data, allowing the use of third-party DLLs on non-Intel architectures.

The 16/32 Loader is responsible for loading, and code and data fixups of binary applications, dynamic link libraries and drivers. The 16/32 loader operates in conjunction with the Virtual Machine and the Thunking Layer to allocate code and data segments, including not-present segments that will be loaded on demand as required by the application. The loader also binds the native Willows Library to the DLL so that calls to the library can be resolved.

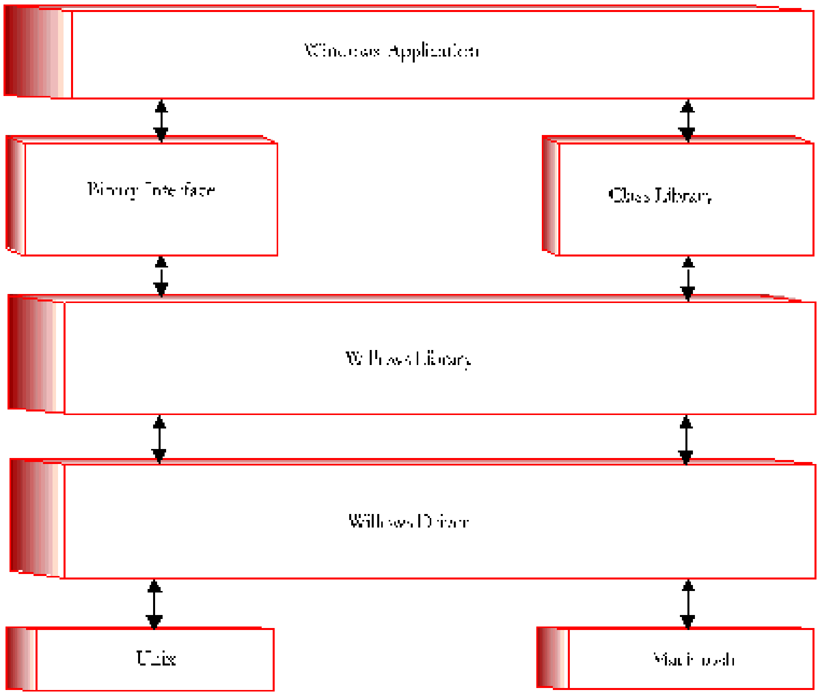
The Willows Library is a platform-independent implementation of the Microsoft Windows API. The library is designed to be portable to a wide range of platforms by isolating platform-specific code from the library and placing it in a platform-specific driver. To further ensure platform independence, the Willows Library is written using the Window API itself, to minimize what functions need to be placed in the driver. (This has been very successful since the drivers in the Willows Library are very small in comparison to Unix and the Macintosh.) This allows the Willows Library to be consistent across multiple platforms, yet compile as native code on each platform to get the best performance possible.

The implementation of the Willows Library is consistent and compatible with the APIW Specification being promoted by ECMA and OSF as a cross-platform Windows API. The library is implemented in the C language and provides both C and C++ compatible binding.

The Willows Library currently supports a 16-bit API interface, but is native internally to the platform on which it executes. This has enabled us to support the body of source and binary applications today, with minimal changes required by source-based applications. As the independent software vendor application base of windows applications migrates to a 32-bit API interface, Willows can easily make the transition with minimal changes to the library. In most cases, it is a matter of modifying the API interface to 32-bits, and using the existing library as it is today. New features, like new controls and common dialogs, can be implemented using the existing API, layered on the current library. Some new features, like threads, can simply be "turned on" given that they are already libraries. Other features, like MAPI and LSAPI, can be implemented as shared libraries, layered on top of the Willows Library.

The Willows Library implements all the most commonly used functions of the Windows API, not including those functions that are specific to running binary applications that are implemented in the Binary Interface. The library supports all of the standard controls and classes, including class styles and messages. The library also supports MDI windows, common dialogs and dialog management functions. The library has built-in support for multitasking applications within a single process, as well as support for the clipboard and DDE. The library maintains properties for each window, and also manages both local and global atom tables. Both native and binary resources are accessible by the library, and can be loaded and unloaded dynamically under application control.

The Willows Library provides a complete set of configuration and debugging features in addition to the Windows API. With the debugging and profiling versions of the library, an application can trace execution of each API and get diagnostic information about the program while it executes. The library provides a built-in debugger that can be used stand-alone or with traditional debuggers to provide detailed real-time information about the state of the application.



The Willows Driver is the platform-dependent portion of the Willows Library. All platform-dependent code for creating and managing windows, drawing on those windows and accessing system-dependent resources resides in this module.

The Driver contains three sub-systems that are accessed by the Windows Library. The sub-systems are:

- Window Manager Interface
-
- Graphics Device Interface
-
- System Services Interface
-

The Window Manager Interface allows the library to manage all the windows that an application creates in a platform-independent fashion. The Window Manager Interface provides such functions as Create and Delete Windows, Resize and Move Windows, Create and Manage Menus on behalf of a window and Set Captions and Icons for minimized windows. All communication with the desktop window manager is done in the platform-specific driver code, and then forwarded to the platform-independent portion of window management. Most external events that the underlying window system provides to graphical applications are captured in this layer as well, and are translated to a uniform set the library can handle. These include messages about creating and deleting windows, as well as messages about window geometry and window redrawing.

The Graphics Device Interface provides the means to actually draw on windows or printers. Functions are available for basic graphic operations such as lines, rectangles and ellipses, as well as more complex operations on graphical objects such as pens, brushes, regions and bitmaps. The Graphics Device Interface has a complete set of functions for text output, from font selection to color and textual special effects such as underlining and strike through**.** The Willows Library and Graphics Device Interface support a wide range of devices, from low color and resolution devices to high color and resolution displays, as well as a wide range of printers.

The System Services Interface provides the platform-dependent code to access the native facilities for file systems, memory allocation, inter-networking and device access. The Willows Library IPC implementation of the clipboard and DDE are supported in this interface. Networking and communications are mapped to their native counterparts in this sub-system. The platform-dependent code associated with dynamic link libraries is implemented in this layer, allowing the Willows Window Library to support the semantics of dynamically loaded libraries on a wide range of platforms.

| Back |
