1. Hylafax

This document does not discuss Hylafax itself, except mention it peripherally. It is intended to document what I have learnt about integrating Faxes and Hylafax into Windows.

2. Windows

All versions of Windows (3.11 for Workgroups) have some sort of Fax capability. I only have to deal with Windows 2000 and Windows XP Pro but I hear that Windows 2003 also work similarly.

Windows provides a fax service, which is like a virtual printer. The printer generates TIFF F-Class (Facsimile class) and then hands the file to a Fax Service Provider (FSP).

Generally an FSP will in turn make use of a TSP (Telephoney Service Provider) to ring up the other fax machine and deliver the file to the other side.

The possibility also exists for a virtual FSP to utilise some other mechanism to get the fax delivered. Email, perhaps. Or Hylafax.

iFax (http://www.ifax.com/)

Linked off the http://www.hylafax.org/links.html[clients] section of the Hylafax.org website is iFax's HylaFSP which does precisely this. It turns out the original dll (and that is all it is) was written by Serge Sandler of Sandler Consulting originally. See <URL:http://members.optusnet.com.au/~hssandler1/hylafsp.htm>.

While this looks like it will do *almost* everything I want, it costs USD$34/machine and I would still want it customised (i.e. default to using the current username, don’t send any kind of password, etc.).
In fact, most of the features touted by HylaFSP are those provided by the Windows Fax Service anyway. So the clever part was writing an FSP which integrated with Hylafax.

3. Developing an FSP

So, I decided if Serge could develop one, I ought to be able to as well. I install the mingw32 tools on my Debian 3.1 machine.

According to <URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/fax/faxusing_5vzm.asp>, the initialise sequence is FaxDevVirtualDeviceCreation is the function is exported and then FaxDevInit (always). There is a header (faxdev.h) which will have the various structures and functions I need too apparently. However it doesn’t exist within MingW32, so I might create it myself based upon the documentation on MSDN.

3.1. faxdev.h

Functions: FaxLineCallback Delivers TAPI events to the fax service provider. FaxSendCallback Notifies the fax service that an outbound fax call is in progress. FaxDevAbortOperation Requests that the fax service provider terminate the active send or receive fax operation. FaxDevEndJob Closes the active fax job gracefully. FaxDevInitialize Initializes the communication between the fax service and the fax service provider DLL. FaxDevReceive Signals a fax service provider DLL that there is an incoming fax transmission. FaxDevReportStatus Queries a fax service provider DLL for status information about an individual active fax operation. FaxDevSend Signals a fax service provider DLL that it must initiate an outgoing fax transmission. FaxDevShutdown Performs cleanup for the fax service provider DLL when the service is shutting down. FaxDevStartJob Indicates the beginning of a fax job or operation, and provides an opportunity for the fax service provider to perform call setup. FaxDevVirtualDeviceCreation Enumerates virtual fax devices for the fax service. FaxUnregisterServiceProvider Unregisters a fax service provider.

structures FAX_DEV_STATUS FAX_RECEIVE FAX_SEND


memory must be allocated on the heap on the heap handle passed in when we are initialised (passed in via FaxDevInitialise)

the FSP must be multithreaded (i.e. proper locking for globals)
FSP receive only Unicode (wide) strings

FSP must call SetLastError as appropriate

Registration/dsregistraton of FSP is via FaxServer.RegisterDeviceProvider / FaxServer.UnRegisterDeviceProvider

apt-get install mingw32

4. Resources