December 4, 1995

Welcome to the Async Professional for Delphi Trial-Run Program! The TRP provides you with the fully functional Async Professional for Delphi serial communications component library. The only limitation imposed is that applications you design using the Async Professional for Delphi TRP will run only while Delphi is also running. This document describes the installation process, and introduces you to the Async Professional for Delphi components. It contains the following sections:

Best of luck with Delphi and the Async Professional for Delphi TRP.

Kim Kokkonen
President, TurboPower Software Co.

1. Component Overview

Borland's Delphi with its Visual Component Library makes it remarkably easy to design slick, professional looking Windows applications, but it doesn't provide any tools for performing serial communications tasks. Async Professional for Delphi fills this gap with a comprehensive set of serial communications components with the following features:

- comport management (open, configure, close, etc.)
- input/output properties and methods
- automatic hardware and software flow control
- events for received data, matched received data, line errors, modem status and timers
- built-in debugging features (I/O tracing and low-level dispatch logging)
- an easy-to-use terminal window component
- automatic ANSI emulation
- Xmodem and Ymodem file transfer protocols
- Zmodem file transfer protocol with crash recover and 8K block option
- Kermit file transfer protocol with sliding windows and long block options
- CompuServe B+ file transfer protocol with 2K block option
- built-in protocol status dialog showing the progress of file transfers
- built-in protocol logging component
- modem database with configuration commands for 100+ modems
- modem control component for configuring, dialing and answering a modem
- phonebook and dialer components to use with the modem and modem database

2. Ordering Information

The list price for Async Professional for Delphi is $179. Owners of TurboPower's Async Professional or Async Professional for C/C++ can upgrade to Async Professional for Delphi for $119. Existing TurboPower customers can take a 20% customer discount from the list price. Owners of Async Professional for Windows can upgrade to Async Professional for Delphi for free.

Async Professional for Delphi includes a 60 day money back guarantee, no royalties for distributing applications based on its components, full source code, free technical support, and free maintenance patch downloads.

For more information, or to order, call 1-800-333-4160 from 8 a.m. to 6 p.m. Mountain time, or write to TurboPower Software, P.O. Box 49009, Colorado Springs, CO 80949. International customers call 719-260-9136. Send faxes to 719-260-7151. Send electronic mail to CompuServe 76004,2611. Visit our CompuServe area in PCVENB section 6. We accept MasterCard, Visa, American Express, Discover, a check in U.S. funds drawn on a U.S. bank, or COD (within U.S. only).

3. Installation

APDTR.EXE is a self-extracting archive. Copy this file to a directory on your hard disk and, from within that directory, enter "APDTR" at the DOS command line. This will extract:

- all of the Async Professional for Delphi units and a registration source file
- this document, README.WRI (a Windows Write file)
- several example demonstration programs that show how to use some of the components
- interface sections for all of the component units

To keep APDTR.EXE as small as possible the help files are distributed in a separate archive. If you wish to add complete online help for for Async Professional for Delphi Trial-Run to your Delphi environment then download APDHLP.EXE from the same place you found APDTR.EXE. Copy that file to a directory on your hard disk and, from within that directory, enter "APDHLP" at the DOS command line. This will extract:

- a complete help system ready to install for use within Delphi

Installing Integrated Help
To install Async Professional for Delphi's integrated help, install the multi-help keyword file by running Delphi's HelpInst utility (do not do this while Delphi is running). Select File|Open from the menu and open DELPHI.HDX. in the DELPHI\BIN directory. A list of the current *.KWF files is shown. Select the Keywords|Add option from the menu and add the APD.KWF file. Select File|Save and exit HelpInst. Add the followin line to both the \WINHELP.INI and MULTIHLP.INI files. These files can be found in your Windows directory. (Replace C:\APD with the full path to the directory where you extracted the Async Professional for Delphi files.):

```
APD.HLP=C:\APD
```

Component Installation
Before you can try out the example programs, you must install the Async Professional for Delphi components into the Delphi palette and rebuild the component library. From within Delphi, select "Options|Install components" from the main menu, select "Add" from the available buttons at the right of the dialog, enter APDREG, and select the OK button. Before closing the "Install components" dialog, append the path name where you extracted the Async Professional for Delphi TRP files to the "Search path" edit control (if it isn't already there). Select OK to have Delphi compile and add the Async Professional for Delphi components to the component library.

APDREG.PAS contains one registration procedure that registers all the components contained in this package. When installation is complete, a new tab called APD appears on the component palette from which you can select the Async Professional for Delphi components.

4. Getting Started

This section provides an overview of all the Async Professional for Delphi components. It shows the icon that appears on the Delphi component palette, the name of the component, its class name, and the unit name that contains the component.

ComPort Component (TComPort, ADPORT)

The comport controls the physical serial port. It has properties for setting line parameters (baud, parity, etc.), hardware and software flow control, and all other aspects of serial port configuration. It has a variety of methods for sending and receiving data in various formats (as single characters, strings or blocks).

Events are provided for any received data, specific received data, line and modem status events and for timed events.

The comport is the basic building block for all other communications components. Protocol, modem and terminal components require a comport before they can be used. When these components are dropped on a form they search for an existing comport component and form a link to it when found. If a comport component is dropped on the form after these other components, the comport broadcasts its presence, allowing other components to establish their required links.

Emulator Component (TEmulator, ADTERM)

The emulator component is usually used in conjunction with a terminal component. It interprets incoming data as either data or ANSI terminal emulation commands.

Terminal Component (TTerminal, ADTERM)

The terminal component, derived from a TWinControl, uses a comport component to send and receive data. It displays all data received by the comport and sends all keyboard data to the comport to be transmitted. The terminal component has optional terminal emulation support, an adjustable scrollback buffer, a facility to capture incoming data to a file, automatic scrollbars and clipboard support.

B+ Terminal Component (TBPTerminal, ADTERM)

CompuServe's B+ file transfer protocol is a bit different from other protocols in that the initial negotiation takes place in terminal mode, potentially intermixed with normal terminal data. The B+ terminal component watches for these negotiation packets and responds automatically. This adds a small amount of overhead so use the B+ terminal component only if the application is intended to be used with CompuServe.

File Transfer Protocol Component (TProtocol, ADPROTCL)

The protocol component provides support for all file transfer protocols (Xmodem, Ymodem, Zmodem, Kermit, B+ and ASCII). It has a large number of general properties and methods that apply to all protocols and a few properties and methods specific to one protocol. All protocol specific properties and methods start with the protocol name. For example, ZmodemRecover and KermitMaxLen apply to Zmodem and Kermit, respectively.

Protocol transfers are background processes. After setting the protocol's properties as desired, the transfer is started by calling either StartTransmit or StartReceive; control returns immediately to the caller and the protocol continues in the background. The protocol communicates with your application through the protocol component's events: OnProtocolStatus, OnProtocolLog, OnProtocolAccept, OnProtocolError and OnProtocolFinish.

Protocol Logging Component (TProtocolLog, ADPROTCL)

A typical protocol operation is to log, usually to a history file, the start and completion of all file transfers. This provides a record of all files sent and received during an unattended period, often with performance information and error codes for failed transfers. The protocol logging component provides this service automatically. If the protocol component finds a protocol logging component on the form, it calls it automatically for OnProtocolLog events.

Protocol Status Component (TProtocolStatus, ADPSTAT)

Another typical protocol operation is to display information about the progress of the transfer; this usually includes elapsed time, bytes transferred, and a progress bar to show the progress of the transfer. The protocol status component provides this service automatically. If the protocol component finds a protocol status component on the form it calls it automatically for OnProtocolStatus events.

The automatic linking behavior of protocol, protocol logging and protocol status components makes it easy to provide full-featured file transfer protocols with very little programming. Just dropping all three components on the form is all that is required (well, at some point the program needs to call StartTransmit or StartReceive, but that's about it).

INI Database Component (TIniDB, ADINIDB)

The INI database component does not yet have a custom bitmap. The INI database component extends the capabilities of INI files by adding a record structure and a key field.

Modem Database Component (TModemDBase, ADMODDB)

The modem database component is derived from the INI database component. It defines a standard set of fields for controlling a modem (reset command, dial command, expected responses, and so on). This component is used to load and use Async Professional for Delphi's database of 100+ modems. The modem database is named AWMODEM.INI.

Modem Component (TModem, ADMODEM)

The modem component, usually used in conjunction with the modem database component, configures and controls the modem attached to your serial port. The modem component has default values for all operations, but is usually loaded from one of the entries in the modem database. The modem has methods for resetting, configuring, dialing and answering a modem.

PhoneBook Component (TPhoneBook, ADPBOOK)

The phonebook component is derived from the INI database component. It provides a very simple (just name and number) INI-file based phonebook for storing commonly called numbers.

PhoneBook Editor Component (TPhoneBookEditor, ADPBEDIT)

The phonebook editor is used in conjunction with the phonebook component. It displays a list of all entries in the phonebook and provides other dialog boxes for adding, changing and deleting phonebook entries.

PhoneNumberSelector Component (TPhoneNumberSelector, ADGETNUM)

The phone number selector is also used in conjunction with the phonebook component, providing a dialing prompt box and history list for the last-dialed numbers.

Modem dialer Component (TModemDialer, ADDIAL)

The modem dialer component does not yet have a custom bitmap. The modem dialer, using a modem component, dials a phone number and displays the progress of the dial attempt (dialing, waiting, connected, and so on).