Appendix A

Software resources for this book

(This appendix is the one that appeared in the printed book. A considerably enhanced one can be found in the file appa.pdf.)

A.1 Source code for the programs

The software that accompanies this text was originally developed in Modula-2, based to some extent on the Pascal code used in Terry (1986). It was subsequently converted to Turbo Pascal, and to C++. Although C++ code is used for most of the illustrations in the text, highly self-consistent source code in all three languages is to be found on the IBM-PC compatible diskette that accompanies the book, along with language-specific implementation notes. The software is also available in other formats - see section A.4.

The C++ source code was mainly developed under MS-DOS using Borland C++ 3.1. It has also been successfully compiled and tested under Linux, using G++, the GNU compiler.

The Turbo Pascal source code was developed to run on any version of Turbo Pascal from 5.5 onwards. However, it makes little use of OOP extensions.

The Modula-2 source code should be immediately usable on PC-based systems using the shareware compiler marketed by Fitted Software Tools (FST), the Stony Brook Modula-2 compiler marketed by Gogesch Micro Systems, Inc., or the TopSpeed Modula-2 compiler developed by Jensen and Partners International (JPI) and now marketed by Clarion Software. It will also compile unchanged under Gardens Point Modula-2 on a wide range of systems.

A.2 Unpacking the software

The software on the diskette is supplied in the form of compressed, self-extracting MS-DOS executable files. There are eight of these files

```
COMMON.EXE - language independent files
CSOURCES.EXE - sources written in C++
PSOURCES.EXE - sources written in Turbo Pascal
MSOURCES.EXE - sources written in Modula-2
FILEIO.EXE - support library for Modula-2 sources
COCORC.EXE - Coco/R for C/C++
COCORP.EXE - Coco/R for Turbo Pascal
COCORM.EXE - Coco/R for Modula-2
```

To unpack the software, simply follow the following steps. Example MS-DOS commands are shown; these may need slight alteration depending on the configuration of your computer.

Windows users may follow an equivalent sequence of operations from within the File Manager.

- Make a backup copy of the diskette and keep the original in a safe place.
- Create a directory to act as the root directory for your chosen sources, for example:

```
MKDIR C:\SRCES
```

- Log onto this as the working directory:

```
CD C:\SRCES
```

- Copy the chosen source file to this directory:

```
COPY A:\CSOURCES.EXE C:\SRCES
```

- Unpack the sources:

```
CSOURCES.EXE
```

- Also unpack the language independent files to the same directory

```
COPY A:\COMMON.EXE C:\SRCES
CSOURCES.EXE
```

  This will create a small directory hierarchy under the `C:\SRCES` directory, in which various subdirectories will appear, usually one for each chapter. For example, you will the find the source code for the programs in Chapter 10 in the directory `C:\SRCES\CHAP10\CPP` (for the C++ versions) or the directory `C:\SRCES\CHAP10\MODULA` (for the Modula-2 versions). Once you have unpacked the system, read the file `C:\SRCES\README.1ST` for further details on how the directories are laid out.

  You may unpack all three of the language specific sources into the same directory tree if you wish - the C++, Modula-2 and Pascal sources are stored in separate directories under the directory for each chapter.

- Create a directory to act as the root directory for your chosen version of Coco/R, for example:

```
MKDIR C:\COCO
```

- Log onto this as the working directory:

```
CD C:\COCO
```

- Copy the chosen version of the Coco/R package to this directory:

```
COPY A:\COCORC.EXE C:\COCO
```

- Unpack the Coco/R system:

```
COCORC.EXE
```

  This will create a small directory hierarchy under the `C:\COCO` directory, in which various

subdirectories will appear, containing the various components of Coco/R. Once you have unpacked the system, read the file `C:\COCO\README.1ST` for further details on how the directories are laid out, and how to complete the installation of Coco/R so that it can be executed easily.

The self-extracting files on the diskette were compressed and packed using the freely available program LHA.EXE developed by Haruyasu Yoshizaki. In terms of the distribution agreement for this program, the complete package for LHA.EXE is itself supplied as a self-extracting executable, LHA213.EXE. You are quite welcome to unpack this file as well, although it is not needed for the operations described above. Further to comply with the distribution agreement, the copyright notice for this package is printed below

```
4. Our distribution Policy

   This software, this document and LHA.EXE, is a copyright-reserved free prog
   and distribute this software free of charge under the following conditions.

1. Never change Copyright statement.

2. The enclosed documents must be distributed with as a package.

3. When you have changed the program, or implemented the program for other OS
   must specify the part you have changed.  Also make a clear statement as to
   address or phone number.

4. The author is not liable for any damage on your side caused by the use of t

5. The author has no duty to remedy for the deficiencies of the program.

6. When you are to distribute this software with publications or with your pro
   the copyright statement somewhere on the disk or on the package.  You canno
   software with copyprotected products.
```

A.3 The compiler generator Coco/R

The compiler generator Coco/R used in this book was originally developed in Oberon by Hanspeter Mössenböck, who also did a port to Modula-2 for the Apple MacMeth system. A further port was done to TopSpeed Modula-2 by Marc Brandis and Christof Brass. This was refined and extended by the author in conjunction with John Gough and Hanspeter Mössenböck, to the point where a single version runs on most Modula-2 compilers available under MS-DOS, as well as the Mocka and Gardens Point compilers available for Unix (and other) systems, including Linux and Free BSD.

A port of Coco/R to Turbo Pascal was done by the author in conjunction with Volker Pohlers.

Coco/R was ported to C by Francisco Arzu, yielding a version that can generate either C or C++ compilers.

The Modula-2 version of Coco/R is supplied as shareware, and is free to academic sites. Other users should contact Professor Mössenböck at the address below to make licensing arrangements.

Prof. Hanspeter Mössenböck
Institute of Computer Science
University of Linz

Alternbergerstr 69,
A-4040 Linz, Austria
Tel: +43-732-2468-9700
e-mail: moessenboeck@ssw.uni-linz.ac.at

A.4 Obtaining the software with ftp

Source code for the programs in this book, and various related other files of interest are available by anonymous ftp from the author's server site

```
ftp://cs.ru.ac.za/pub/languages.
```

Look for the file READ.ME for details of what to get, how to unpack the files, and for differences from the software on the diskette.

The latest versions of Coco/R for a variety of languages and operating systems should be available from the following servers

Europe: `ftp://ftp.ssw.uni-linz.ac.at/pub/Coco`
USA: `ftp://ftp.psg.com/pub/modula-2/coco`
Central America: `ftp://uvg.edu.gt/pub/coco`
Australia: `ftp://ftp.fit.qut.edu.au/pub/coco`
South Africa: `ftp://cs.ru.ac.za/pub/coco`

Look for the files `README.1st` and `READ.ME` for details of what to get and how to unpack the kits.

The original report on Coco/R (Mössenböck, 1990a) can be obtained from

```
ftp://ftp.ssw.uni-linz.ac.at/pub/Papers/Coco.Report.ps.z
ftp://cs.ru.ac.za/pub/coco/Coco.Report.ps.z
```

The PCCTS compiler construction kit mentioned in Chapter 10 is available from

```
ftp://ftp.parr-research.com:/pub/pccts
```

`mtc`, the Modula-2 to C translator program mentioned in Chapter 2 is available by anonymous ftp from

```
ftp://ftp.psg.com:/pub/modula-2/grosch/mtc.tar.Z
ftp://ftp.ira.uka.de:/pub/programming/cocktail/mtc.tar.Z
```

`p2c`, the Pascal to C translator program mentioned in Chapter 2, and `cperf`, the perfect hash function generator mentioned in Chapter 14, are available by anonymous ftp from any of the sites that mirror the Free Software Foundation GNU archives. The primary server for these archives is at `prep.ai.mit.edu`. Among many others, the Linux sites, such as those at `tsx-11.mit.edu`, `sunsite.unc.edu` and `src.doc.ic.ac.uk` also carry copies of the GNU archives.

Freely available early versions of the Cocktail compiler construction tools mentioned in Chapter 10 may be obtained by anonymous ftp from

```
ftp://ftp.ira.uka.de/pub/programming/cocktail
ftp://144ftp.info.uni-karlsruhe.de/pub/cocktail
```

For the commercial version and support, contact Josef Grosch by email at grosch@cocolab.sub.com

Versions of the Gardens Point Modula-2 compiler for DOS, Linux and FreeBSD are available from

```
ftp://ftp.fit.qut.edu.au/pub/gpm_modula2
ftp://ftp.psg.com/pub/modula-2/gpm
```

Versions of the Mocka Modula-2 compiler for Linux and FreeBSD are available from

```
ftp://144ftp.info.uni-karlsruhe.de/pub/mocka
```

The shareware FST Modula-2 compiler for MS-DOS systems is available by anonymous ftp from

```
ftp://ftp.psg.com/pub/modula-2/fst/fst-40s.lzh
ftp://cs.ru.ac.za/pub/languages/fst-40s.lzh
```

In case of difficulty, consult the author at the address given below.

Pat Terry
Computer Science Department,
Rhodes University
GRAHAMSTOWN 6140, South Africa
Tel: +27-46-6038292
e-mail: cspt@cs.ru.ac.za

A.5 The input/output module FileIO

When this book was first published, standardized I/O for Modula-2 was not yet widely available (and was incompatible with extant Modula-2 compilers). The Modula-2 source code on the diskette attempts to get around this problem by providing (another!) I/O module, called `FileIO`. The definition module for `FileIO` is acceptable to all the compilers mentioned above; implementations have been supplied for each that differ internally only in a few places.

On the diskette you will find a self-extracting file `FILEIO.EXE` that contains the sources of `FileIO` for a variety of MS-DOS compilers. You will need to install the version of FileIO that matches your compiler.

- Make a directory to contain these sources:

  ```
  MD C:\FILEIO
  ```

- Log onto this as the working directory:

  ```
  CD C:\FILEIO
  ```

- Place the source diskette in the A: drive and unpack FILEIO.EXE.

  ```
  A:\FILEIO.EXE
  ```

This will create a small directory hierarchy under the `C:\FILEIO` directory, in which various subdirectories will appear, one for each compiler.

In the `C:\FILEIO` directory you will find the definition module `FILEIO.DEF`, and in a subdirectory of `C:\FILEIO` you will find the implementation module `FILEIO.MOD`. You will need to proceed as follows, on the assumption that you have a "working" directory `C:\WORK` in which you normally develop programs (or, preferably, install `FileIO` in the library directory or directories for your Modula-2 compiler).

```
CD C:\WORK
COPY C:\FILEIO\FILEIO.DEF
COPY C:\FILEIO\xxx
```

where xxx =

`JPI` (TopSpeed compilers)
`FST` (Fitted Systems Tools compilers)
`LOG` (Logitech compilers)
`STO` (StonyBrook compilers)
`GPMPC` (Gardens Point PC compiler)

Follow this by compiling `FILEIO.DEF` and `FILEIO.MOD`.

`FileIO` provides the usual services for opening and closing text files, and for reading and writing strings, words, whole numbers and line marks to such files. It can also handle random access binary files, as block read and write operations are provided. In addition there are some utility procedures, for obtaining command line parameters and environment strings, and for the output of dates and times. The module is of fairly widespread applicability beyond the confines of this text, and is compatible with the modules generated by Coco/R (which assumes the module to be available). As an example of a library module it is really rather too large, but has been developed in this way to minimize the number of non-portable sections and modules needed for implementing the programs in the book. The sources supplied will act as models of implementations for compilers not mentioned above. In case of difficulty in this regard, please contact the author.