The Met.Office

UNIFIED MODEL DOCUMENTATION PAPER NO X0

UNIFIED MODEL FOR EXTERNAL USERS

by

A. Van der Wal

Version 3

29 February 2000

Model version 4.5 (beta release)

Numerical Weather Prediction
The Met. Office
London Road
BRACKNELL
Berkshire
RG12 2SZ
United Kingdom

Modification Record		
Document Version	**Author**	**Description**
1	N. Farnon	Original Version
2	A. Van der Wal	Update for vn4.4 external (beta) release
3	A. Van der Wal	Update for vn4.5 external (beta) release

Contents

1 Introduction

The major component of the enclosed tape or CD-ROM is the Unified Model (UM). The Fortran code and UNIX scripts, is an enhanced version of the vn4.5 code used by The Met. Office for climate modelling and operational weather prediction. The code, scripts and data supplied have been tested on a Cray T3E machine, a Silicon Graphics system, a Dec Alpha workstation and a PC running Linux. The system must support Fortran90 and ANSI C along with the IEEE standard for data representation. The PC requires a Fujitsu Fortran90 compiler to be installed.

The Unified Model User Interface (UMUI) is an upgraded version (4.5.1) which uses Tcl/Tk 8.0. Information on how to set up the UMUI is supplied in a README file on the tape. The UMUI itself has detailed help windows for new users.

Also supplied are two graphical utilities – xcumf and lampos. Xcumf allows visualisation of UM dumps, ancillary files, and model output, and can be used to plot differences of two fields. It is intended to be used as a first-look tool, and field manipulation is not possible. Lampos is used to determine the grid, co-ordinates and rotated pole of a limited area model, which can be supplied to the UMUI.

Running the model in global operational weather prediction mode is not possible with the general release. However, interested users may contact the Head of Numerical Weather Prediction at The Met. Office if more information is required.

It is envisaged that the Unified Model system and the UMUI will be unpacked and built using a userid specifically set up for the UM. Individual users can then upload and run experiments under their own userid.

2 Supplied Software and Data

2.1 Choice of Media

In the UM package you should have received software and data on either a 90 metre DAT or two CD-ROMs. Please follow the instructions for the media you have chosen; these are given in sections 2.2and 2.3

2.2 Instructions for Tape

2.2.1 Files Contained on the Tape

The following files are available from the tape:

PACKAGE	FILENAME	INSTALL DIRECTORY	DESCRIPTION
UMUI	umui_package.tar	UM Userid	Tcl/Tk, GHUI2.0a1 and UMUI2.0 for installing the UMUI
	umui_templates.tar	UM Userid or users own userid	Template files and scripts for use with the UMUI
UM System	um_system.tar	UM Userid	UM vn4.5 plus initial data & modifications for supported experiments
	umui_input.tar	Users own userid	User STASHmaster files and script inserts
	data32.tar	UM Userid	Start dumps, ancillary files and boundary conditions at 32-bits
GCOM	gcom.tar	UM Userid	GCOM libraries, required if running MPP experiment
Lampos, xcumf & xconv	um_visual.tar	UM Userid or users own userid	X-Windows based tools for positioning and sizing a Limited Area Model, and plotting UM output fields, dumps and ancillary files
Documentation	um_doc.tar	UM Userid	Full documentation system

Table 1

2.2.2 Retrieving Files from Tape

Before extracting the data you may need to know the device name of your DAT drive; if there is just one connected to your system then the tar command may be able to access it by default (without specifying a drive name). Make sure that the drive does not perform byte swapping on extraction of the data.

At this point, it is for the user to decide whether to extract all the tarfiles from tape or one by one, as they are required. You should note where the files are to reside, by referring to Table 1. It is possible to extract all files from tape into the HOME directory of the UM Userid and then transfer the relevant ones into the users own

account.

To extract all files from tape, issue the command:

tar xvof $DEVICE

where $DEVICE is the name of your tape drive (something like /dev/rmt/0m). Alternatively, you can try:

tar xvo

which should reference the default tape drive.

To extract a single file, type:

tar xvof $DEVICE $filename

where $filename is the name of the file you require, e.g. umsystem.tar. To untar any of the files extracted from tape, type

tar xvof $filename

2.3 Instructions for CD-ROM

2.3.1 Files Contained on CD-ROM

CD-ROM one contains the UMUI, the UM system and GCOM packages together with 64-bit input data, UMUI experiments and some graphical utilities. CD-ROM two contains the UM Documentation System (UMDS), plus 32-bit input data. See Table 1 for details of the filenames. Because the UMDS can be used directly from the CD, it is not a tarfile, but a complete directory structure. Therefore, if you wish to copy to hard disk, it is probably easier to create a tarfile, usind the command:

tar cvf um_doc.tar umdoc_system

and then copy the tarfile to hard disk.

2.3.2 Retrieving Files from CD-ROM

You may first need to mount the CD-ROM drive on your system. This often requires root permission. Speak to your system administrator if you are unsure how to do this.

You can browse the files on CD-ROM directly by moving to the directory where the CD-ROM has been mounted. This is often */cdrom* or */CDROM*. The files in here are all those which are on the CD. You will need to copy the tarfiles onto hard disk. The UM documentation system, which is in HTML format, is suitable for browsing using Netscape Navigator or Internet Explorer (on Windows Platforms). The UMDS can be

started up by pointing your browser at the file index.htm. If your CD-ROM is mounted as /cdrom, this can be achieved by entering the following address into the "Location" window:

file:/cdrom/umdoc_system/index.htm

3 The Unified Model Software

3.1 Installation of the Unified Model

The installation procedure is described in detail in Documentation Paper X4. The supplied scripts build a replica of the software system which runs on the CRAY T3E computer installed at The Met. Office.

Although it is expected that the installation should proceed without too many hitches, there are a few points which, if noted, will probably smooth the way:

- Since there is no internationally agreed standard for the Fortran to C interface, the defaults provided in this release may need to be changed. The options available are discussed in Section 4 of Documentation Paper S5. A utility called fc_test is provided for determining the interface on your platform, and this will run automatically when the UM system is installed.

- The Fortran representation of INTEGERs, LOGICALs and REALs must all have the same word length - either 32 or 64 bits long.

- There is no standard for the wordlength associated with the C types *int, long, float* and *double*. The appropriate C type must be used to interface with Fortran REALs. How this is done is discussed in section 4 of Documentation Paper S5. The utility fc_test is provided to find the wordlength on your platform.

- Only the IEEE and Cray standards for the representation of floating point numbers are supported. 64-bit data files are provided as well as a utility to swap between the big-endian and little-endian ordering of bytes in each word.

- Automatic arrays are used throughout the code in order to manage the use of workspace. It is expected that your system will be able to deal with this aspect of dynamic memory allocation. This is part of the Fortran90 standard, but not Fortran77 (although many, if not all, implementations do support it).

- If changes to the source code are found to be necessary, it is recommended that they are introduced via the *nupdate* utility. In this way, any suggestions for enhancing the portability of the code can be incorporated into future releases of the model with the minimum of effort. For debugging, it is often easier to edit the extracted Fortran and C files directly, but any changes will be lost if a new compilation via the UMUI is performed, unless the switch SKIP_SRC_EXTRACT (in SUBMIT) is set to true.

- The code makes widespread use of namelists; in particular the code assumes multiple namelists per file and that REWIND can be used to re-read a namelist. Problems may be encountered with the spacing occurring before the & declaring the beginning of a namelist. As there is no international

standard, some platforms require namelists to finish with an &END, others require a /. Within the UM, files containing namelists frequently use # to start comment lines. This is not acceptable on some platforms, in which case, the removal of all comments is required before the namelist file can be used.

- The Unix “banner” command is not available on some platforms. As the UM is unpacked, a test will be performed, and if “banner” doesn’t exist, the figlet utility will be installed and renamed “banner”.

- The Unix utilities awk, grep and sed are sometimes not compatible with the UM. So far, this has only been the case on Sun platforms which aren’t supported in this release. For this reason, calls to these utilities have been replaced with environment variables, UM_AWK, UM_GREP and UM_SED, which are set in unpackmodel and can easily be altered by editing the values in setvars.

3.2 Unified Model Experiments

Three experiments are provided in the release: two atmosphere experiments (one global, one limited area) and one ocean experiment. It is recommended not to change the experiments supplied other than 'personal details' until each experiment has successfully completed as is.

Additionally, a coupled model can be provided on request, although this distribution will be restricted. It can only be run on a T3E, and The Met. Office retains the right to refuse any applications. Instructions on how to convert the coupled model to an ocean-only model will be provided.

The experiments already reside in the database of the UMUI, under the userid “frav”, and are therefore “read-only”. Those which are required by the user should be copied, enabling write permission. Details of the experiments are as follows:

- **PUM Atmosphere – HadAM3 AMIPII**
  This is a 1920 timestep (40 model days) Atmosphere Climate Model (96*73*19 gridpoints). The formulation is that of HadAM3.

- **PUM Ocean – HadOM2**
  This is a 240 timestep (10 model days) Ocean Model (98*73*20 gridpoints). The formulation is that of HadOM2.

- **PUM Limited Area – M20**
  This is a 288 timestep (1 model day) Limited Area Model (92*92*31 gridpoints). The formulation is that used by the UK Operational Mesoscale Model, cycle M20.

Within each experiment directory there are 3 jobs:

#a - non-MPP

#b - MPP generic
#c - MPP T3E

3.3 The UM Compile System

The compile system relies extensively on the UNIX "make" facility. Decks are extracted (by nupdate) into individual Fortran files (different from the old system where all the Fortran code was concatenated into one large file) and a Makefile is automatically created. This is done, not only for the main UM run, but also for the small executables and pre-compiled sections. It is therefore easy to see which decks have been used to create each executable, and changes can be tested quickly by editing the Fortran code directly. It is advisable that all changes should eventually be gathered into a modset for use with nupdate. Here are examples of some of the occasions when a user may find it easier to work with the extracted .f (and one .c) files directly:

- Adding WRITE or PRINT statements to find the value of certain variables.
- Adding a call to FLUSH (or your system equivalent) to flush the buffer at certain points in the code.
- Compiling some files with different options, especially -g for debugging
- Using UNIX "grep" to find occurrences of strings. This can be particularly useful to find out where UM/small executable error messages come from.

For full details of the compile system, see UMDP Z52.

3.4 UM output

3.4.1 General Details

Three forms of output file are produced by the UM: "job log" files, "fieldsfiles" and "dumps". The UM utilities pumf and cumf can be used to print out information from fieldsfiles and dumps. See UMDP F5 for more details.

3.4.2 Log Files

The log file is often known as the "output" from the model run. It contains anything that is echo'd (UNIX) or written to stdout or stderr and is the place to look to see if your job has completed successfully. Whilst your job is running, most of the output is sent to a file in your $DATAW directory called $RUNID.out. This file is updated constantly. In the case of MPP runs Fortran unit 6 output (stdout) is sent to $RUNID.fort6.pe0. When the model reaches completion, the contents of the file(s) $RUNID.out (and $RUNID.fort6.pe0) are moved to $UM_HOME/umui_out/unique_filename, where unique_filename is of the form:

xaadd000.xaadd.d98281.t162219.leave.

In this case,
xaadd000 = jobname as specified in the UMUI (usually ${RUNID}000)
xaadd = $RUNID

d98281 = d{2 digit year}{3 digit day of year}
t162219 = t{2 digit hour}{2 digit minute}{2 digit second}

3.4.3 Fieldsfiles

Prognostic and diagnostic fields can be output at intervals throughout the integration. These are chosen from the STASH panels of the UMUI and are sent to files attached to Fortran unit numbers. These are called fieldsfiles and contain header and data information. The filename depends on the time convention you are using and whether or not you are asking for periodic re-initialisation of ppfiles. Files with fixed names are written to $DATAW; those with variable names, generated using model time information, are written to $DATAM. See UMDP 7 for more information on filenaming conventions.

The utility convpp can be used to convert these fieldsfiles into sequential ppfiles. A simple Fortran program can be written (see Appendix A) to read in the data, with each field having a 64-word header followed by the data, and it can be written out in a format required by your preferred visualisation package. See UMDP F3 for information on the format of data files.

3.4.4 Dumps

The dump files can be used as start dumps for model runs, and are usually kept for continuation runs, particularly for long model integrations. They are output to your $DATAM directory.

4 Getting Started with the UMUI

4.1 General Details

The UMUI supplied for external users is version 2.0

Documentation on the UMUI may be found in The Met. Office Unified Model Users Guide. (Chapter 4.1). The ’umsubmit’ utility mentioned in section 4.1.7 of the UM User Guide (UMUG) is supplied in the directory umui_templates, but may need modification for your platform. Please be sure to read the UMUG before proceeding with making any changes to the supplied UMUI jobs.

The UMUI distribution, which includes Tcl/TK, GHUI and UMUI is installed by following the instructions in the README files included with the distribution. Also, see chapter 8 of UMDP X4.

A description of how to create experiments and jobs, and change some details of your job through the UMUI panels, and the equivalent hand-edits, is included in the following sections. Hand-edits are useful for small changes when re-processing your job may not be appropriate. See UMUG, section 4.1.7 for further details on hand-editing.

4.2 Starting the UMUI

To activate the UMUI, type ’umui &’.

4.3 Creating an Experiment

The first time you use the UMUI, the experiment table should be empty as you do not yet own any experiments. In this case, create a new experiment by selecting "New" from the "Experiment” menu; enter a few words to describe the experiment. The UMUI will select a 4-letter experiment identifier (EXPTID) automatically. At this stage, your experiment will not contain any jobs.

Alternatively, you can copy a pre-existing experiment. To do this, highlight the experiment which is to be copied, and choose “Copy” from the “Experiment” menu. This will copy all jobs contained in the experiment.

For convenience, the UMUI distribution already contains the standard PUM experiments in its database, under userid “frav”. To make these visible, select “Filter” from the “Search” menu, and then do a filter on the “Owner” frav.

4.4 Creating a Job

The simplest way of creating a new job is to copy an existing one either from your own list or from another user. To do this, click to highlight the experiment in which the job will be created, and also highlight the job to be copied. If copying from

another user, it will be necessary to filter on two userids. Select “Copy” from the “Job” menu, and then add the description of the new job. The new job will be at the same version as the old job.
It is essential that PUM jobs are brought up to version 4.5.1. To do this, click to highlight the new job, making sure that all other jobs and experiments have been unhighlighted. Then choose “Upgrade version” from the “Job” menu, followed by “4.5.1”.

To create a new (empty) job, click on a pre-existing experiment, so that it is highlighted. Now select "New" from the "Job” menu; enter a few words to describe the job. The UMUI will ask you to select a version for the job - choose "vn4.5.1". Next you will be asked to select a job identifier (JOBID). This can be anything from "a" to "z. Once your job has been created, you verify it is there by clicking on the folder next to the EXPTID, which will open to reveal your new job.

If you wish to copy a job from or to a user on a different umui (for instance, if collaborating with someone from The Met. Office), this is done using “basis” files. To download a basis file from your job, highlight the job and select either of the “Open” options from the “File” menu. Next, select “Download” from the menu along the bottom of the window, and enter the directory path and filename to download to.

To upload a basis file, firstly create a job, either from scratch or by copying one that already exists. Open the job as “Open read write” from the “File” menu, and choose “Upload” from the bottom menu, supplying the full path (environment variables and ~ are not allowed) and name of the basis file when required. For safety, “Save” the uploaded job immediately, which will save the details in the UMUI database.

4.5 Pre-run UMUI Checks

Before running, you should check the following UMUI windows and amend if required:

1. Change the userid and the machine you wish to run your experiment on, in the window:

-> Model Selection
--> User Information and Target Machine
---> General Details

and

-> Model Selection
--> User Information and Target Machine
---> Target Machine

2. Check the compiler in the window:

-> Model Selection

--> User Information and Target Machine
---> Target Machine

For jobs a & b, the compiler is set to "Generic Type"; for job c, it is set to "F90 on Cray T3E". These defaults can be changed if required.

3. The directory $HOME/PUM_Output/vn4.5 will be used for output data files from the model run. This can be changed in the window:

-> Model Selection
--> Sub-Model Independent
---> Script Inserts and Modifications

4. Check your submission method in the window:

-> Model Selection
--> Sub-Model Independent
---> Job submission, resources and re-submission pattern

For jobs a & b, it is set to use "at"; job c uses qsub on a Cray platform. These defaults can be changed if required.

5. User STASHmaster files are listed in the window:

-> Model Selection
--> Atmosphere
---> STASH
----> User-STASHmaster files. Diags, Progs and Ancills

These must reside on the platform where the UMUI is installed. Use full pathnames and do not use environment variables, but you can use “~”. You are advised to use ~ followed by your user-id rather than just ~ if others may use the file.

To include the user-STASHmaster files you should change the value of ~frav to your user-id. The user_stash directory will already exist in your $HOME/umui_input directory.

6. Model modifications are listed in the window:

-> Model Selection
--> Sub-Model Independent
---> Compilation and modifications
----> Modifications for the model

We have tried to build any portability mods into the UMPL, but changes to the science have been left as mods applied through this window. This allows easier comparison between experiments from The Met. Office, and those in the PUM release. Any platform specific mods which couldn’t be built into UMPL are also

included here.

7. Script modifications are listed in the window:

-> Model Selection
--> Sub-Model Independent
---> Script Inserts and Modifications

You may need to use this window if you are building on an unsupported platform.

At this point, it is worthwhile selecting “Check Setup” from the bottom menu, which will flag any discrepancies in your job, for instance choosing an MPP scheme in a non-MPP job. These should be corrected, and the job resaved.

When you are happy, the job can be “Process”ed. This will produce a directory called $UM_HOME/umui_jobs/$EXPTID$JOBID which will be populated with UNIX scripts and NAMELISTS for running your job. For ease of reference, $EXPTID$JOBID (a 5-letter value) is often called $RUNID.

4.6 Job Submission

Now that you have tailored your umui job, you can send it to run, using the “Submit” button from the bottom menu or the umsubmit script supplied with the UM.

The files in directory $UM_HOME/umui_jobs/$RUNID will be copied to directory $UM_HOME/umui_runs/$RUNID-$TIMESTAMP on the platform you wish to run your experiment. $TIMESTAMP is a unique numerical value which allows simultaneous running of the same job without the control files being overwritten. A top level script called umuisubmit is created in $UM_HOME/umui_runs/$RUNID-$TIMESTAMP, and it is this which is executed (automatically).

Note: The default jobs are all set up to reconfigure the data, compile the source code to build the executable, and run the executable.

4.7 Changing Specifics of a Job

4.7.1 Reconfiguration Only

There may be occasions when you only require the reconfiguration, without a model run.

a. In the UMUI, the switch for "reconfiguration only" is as follows:

-> Model Selection
--> Sub-Model Independent
---> General Configuration and Control
■ Perform the reconfiguration step only

Obviously, you will need to have the reconfiguration turned on as well. For an atmosphere run, choose:

-> Model Selection
--> Atmosphere
---> Ancillary and Input data files
----> Start Dump
■ Using the reconfiguration

For an ocean run:

-> Model Selection
--> Ocean
---> Input files
----> Start Dump
■ Using the reconfiguration

b. The hand-edit is as follows:

In SUBMIT, change STEP=1 to STEP=99, and in SCRIPT, make sure that RECONA=true for the atmosphere, and RECONO=true for the ocean.

4.7.2 Switching off Reconfiguration

If you intend to run from the same analysis, using the same ancillary files and resolution, more than once, there is no need to have the reconfiguration switched on each time. After the first time, simply use the reconfigured start dump created in your first run. This will save time and computer resources.

a. The reconfiguration switch in the UMUI is set as follows for the atmosphere:

-> Model Selection
--> Atmosphere
---> Ancillary and input data files
----> Start Dump
■ Using the reconfiguration

and for the ocean:

-> Model Selection
--> Ocean
---> Input files
----> Start Dump
■ Using the reconfiguration

b. The hand-edit to turn off reconfiguration is as follows:

In SCRIPT, change RECONA=true to RECONA=false, or in the ocean model, change the value of RECONO.

4.7.3 Executing Compilation Step Only

a. To select compilation only in the UMUI, go to the following panel:

-> Model Selection
--> Sub-Model Independent
---> Compilation and Modifications
----> Compile options for the model
■ Compile and build executable named below, then stop

b. The equivalent hand-edit is:

In SUBMIT, change the value of STEP to STEP=0

4.7.4 Skipping Compilation

a. To skip compilation and run from a pre-built executable, go to:

-> Model Selection
--> Sub-Model Independent
---> Compilation and Modifications
----> Compile options for the model
■ Run from existing executable, as named below

b. The hand-edit for this is:

In SUBMIT, change STEP value to STEP=4

4.7.5 Continuation Run

a. This can not be done through the UMUI

b. The hand-edit is as follows:

In SUBMIT, change TYPE=NRUN to TYPE=CRUN

4.7.6 Prohibiting Conversion of the Compile Directory to a Tarfile

a. This can not be done through the UMUI

b. The hand-edit is as follows:

In SUBMIT, change SKIP_TAR_COMPDIR=true to SKIP_TAR_COMPDIR=false

4.7.7 Skipping the Extraction Step

a. This can not be done through the UMUI

b. The hand-edit is as follows:

In SUBMIT, change SKIP_SRC_EXTRACT=false to SKIP_SRC_EXTRACT=true

This would be required if the Fortran source files in the compile directory had been hand-edited. With SKIP_SRC_EXTRACT=false, the files would be overwritten by re-extracted source.

4.7.8 Building Sections

Although many of the model sections will have been pre-built (see obj_xref sections containing a line like A01_1B BUILD @host@ G0) the lesser used sections have not. If you choose one of the unbuilt sections in your model run, it will fail with a message telling you to build the section. In this case, hand-edit SUBMIT and change BUILDSECT=false to BUILDSECT=true, and re-submit.

5 Data

5.1 Start Dumps

The start dumps provided for the supported experiments are at 64-bits, but they have also been converted to 32-bits and are held in separate directories. The standard 64-bits versions are held in $UMDIR/PUM_Input/vn4.5/dumps and are:

- Apr01_1998.12.mes – used in UK mesoscale expt. xaac, 12Z on 01/04/1998
- Dec01_1994.clim – used in HadAM3 climate expt. xaaa, 00Z on 01/12/1994
- cbtpho.da60910.new.ocean – used in HadOM2 ocean expt. xaab, 00Z on 01/09/1860
- aaxzka.da59c10.coupled – used in HadCM3 coupled expt., 00Z on 01/12/1859
- aaxzko.da59c10.coupled – used in HadCM3 coupled expt., 00Z on 01/12/1859

The atmospheric start dumps can be configured onto any reasonable grid; no interpolation is possible for the ocean model.

The 32-bits dumps have the same filename and can be found in the directory $UMDIR/PUM_Input/vn4.5/dumps32.

If running on a little-endian machine (BIGEND=false), then remember to run the bigend utility to byte-swap all dumps. This utility will have been compiled automatically when you unpacked the UM, and can be found in the directory $UMDIR/bin.

5.2 Ancillary Files

In the sub-directories of $UMDIR/vn4.5/ancil are a number of standard ancillary files which can be incorporated into the model dump via the reconfiguration program. It is important that the ancillary files have the same resolution as your model run. If not, it is necessary to create new ancillary files of the correct resolution and domain. A README file exists in each directory giving a description of the files therein. See UMUG section 4.9 for further information.

Ancillary files are provided at 32-bits in the sub-directories of $UMDIR/vn4.5/ancil32

Extra non-standard ancillaries can be found under $UMDIR/PUM_Input/vn4.5/ancil, and the 32-bit equivalents are under $UMDIR/PUM_Input/vn4.5/ancil32.

If running on a little-endian machine (BIGEND=false), then remember to run the bigend utility to byte-swap all ancillary files. This utility will have been compiled automatically when you unpacked the UM, and can be found in the directory $UMDIR/bin.

5.3 Lateral Boundary Conditions

These are required when performing a limited area run and are provided by the driving model. They can be as frequent as desired, although the user has to decide whether they favour more data and hence larger file sizes in preference to something more manageable. The supported mesoscale experiment has a lateral boundary conditions (LBC) file provided which allows updating every 3 hours. No other LBC's are available, but the user can create their own file by running a global model with production of LBC's switched on. The LBS can be found in $UMDIR/PUM_Input/vn4.5/lbcs with the 32-bit equivalents being in directory $UMDIR/PUM_Input/vn4.5/lbcs32.

If running on a little-endian machine (BIGEND=false), then remember to run the bigend utility to byte-swap all LBC files. This utility will have been compiled automatically when you unpacked the UM, and can be found in the directory $UMDIR/bin.

6 Documentation

6.1 General Details

Complete documentation is available in the file um_doc.tar if you have a tape version of the installation, or on CD-ROM two, in the directory umdoc_system. The top level of the documentation system is written in html, with links to documents which are in postscript format. There is also a source code browsing facility. The system is a self contained package.

6.2 Installation from Tape

Extract the file um_doc.tar from tape using

tar xvf $DEVICE um_doc.tar

where $DEVICE is the name of your tape device (something like /dev/rmt/0m). Next, untar the file using

tar xvf um_doc.tar

The top level directory of the extracted tree is called umdoc_system.

6.3 Installation from CD-ROM

See section 2.3.2.

6.4 Using the Documentation System

To access the documentation, bring up the file index.htm in your browser using the following command:

file:/*my_full_pathname*/umdoc_system/index.htm

if the documentation is installed on hard disk, where *my_full_pathname* is the path where you have installed the UM documentation, or

file:/cdrom/umdoc_system/index.htm

for documentation straight from the CD-ROM.

Before attempting to install the UM it is important that you thoroughly read the following two papers of which hard copies have been provided:

UMDP X0 - Unified Model for External Users (this paper)
UMDP X4 - Guidelines for Building the Unified Model on External Systems

7 Graphical Utilities

7.1 General Details

The UM package includes three x-windows applications - Lampos, which can be used for determining some input parameters of a Limited Area Model (LAM), xcumf, a graphical interface to the cumf utility, which will plot UM fields, and differences between fields, and xconv (courtesy of Jeff Cole at Reading University) which converts UM output into other data formats.

7.2 Lampos

Lampos is an X-Windows based tool for positioning and sizing a LAM. It is designed to make it easy for both novices and experts alike to determine the parameters required for input into the UMUI. More details on the operation of Lampos may be found by browsing help.html (in the Lampos directory) with a Web browser such as Netscape.

Lampos makes use of the Tcl/Tk 8.0 wish shell. There is a Tcl/Tk 8.0 installation included with the UMUI package, which can be used if you don’t already have this version installed on your system. Please check the file read.me in the lampos top-level directory for installation instructions.

7.3 Xcumf

Xcumf allows visualization of PUM fields. It will display output fieldsfiles as well as dumps and ancillary files. It is also possible to produce plots of the difference between two fields, but no other data manipulation is available. It has been written for the T3E, and although it has the potential to run on other platforms, it hasn’t been tested for these. Therefore we can provide support on a best endeavours basis, but much of the work will be down to the user. It has also been noted that some changes are needed for xcumf to work for output from ocean models.

7.4 Xconv

Xconv's main purpose is to convert one data format to another; it allows you to subsample the data in the vertical and time dimensions. Xconv also allows various data transformations before conversion, e.g. spectral to gridpoint transformation, bilinear and area weighted interpolation of horizontal slices, zonal and meridional means. There is also the facility to view horizontal data slices with a block fill type graph or a table of the data values. Xconv outputs data in the following formats:

- NetCDF
- DRS
- UTF

UTF is a, largely obsolete, UGAMP (UK Universities Global Atmospheric Modelling Programme) defined data format, which a few users still use. DRS is a data format

from PCMDI (Programme for Climate Model Diagnostics and Intercomparison), which quite a few people here still use but is no longer supported by PCMDI. NetCDF is the best supported output format and is what is recommended. Xconv can read data in the following formats:

- Various files used by and produced from the UM , i.e. dump, ancillary, boundary data and fields files
- PP format
- GRIB
- NetCDF
- DRS

Xconv cannot handle UM/PP files with timeseries type data. Associated with xconv is convsh, which allows a user to access xconvs facilities in a script file; the scripting language is tcl.

There is a web page available which gives some useful information about xconv. It can be reached at:

http://ugamp.nerc.ac.uk/~jeff/xconv

This page is duplicated in the PUM Documentation System, and can be found via the index.

8 Support Information

Limited technical support is available during installation of UM vn4.5 release for Cray T3E, DecAlpha, SGI Octane/Origin platforms, and PC's running Linux. Science support may be available for those who have established a research collaboration with members of The Met. Office. Requests for data other than that supplied on the tape should be made initially to one of the Portable Unified Model team members. This will require a further agreement with The Met. Office and a charge will be made.

Two mailing lists have been set up for Portable Model users:

- pum-support
- pum-announce.

pum-support can be used by anyone to post general support/problem queries, and anyone can respond to them. pum-announce is for the PUM group here at The Met. Office to post announcements of new releases, fixes, etc.

To subscribe to a group, post a message to majordomo@meto.gov.uk
and in the body of the message write:

subscribe pum-support
or
subscribe pum-announce

depending on which list you want to subscribe to. Unsubscribing is very similar, just replace "subscribe" by "unsubscribe".

To post to the pum-support group, simply post to pum-support@meto.gov.uk

Other useful contacts are listed below:

Portable Unified Model

Anette Van der Wal	avanderwal@meto.gov.uk
	(+44 1344 854098)
Paul Burton	pmburton@meto.gov.uk
	(+44 1344 854972)

Unified Model User Interface

Steve Mullerworth	sdmullerworth@meto.gov.uk
	(+44 1344 856899)

Unified Model Librarian

Mike Hatton	mjhatton@meto.gov.uk
	(+44 1344 856485)

Appendix A

The following is a simple Fortran program which will read in UM fieldsfile data which has been converted to "pp" format using the UM utility convpp. Note that because the u and v (and any related fields such as surface stress) fields are on a different grid to the main fields (1 less point in the N->S direction), the pp-header information is used to set up the array size of the data to be read in.

```
      PROGRAM READPP

      INTEGER             IHEAD(45)
      REAL                RHEAD(19)
      REAL                DATA1(7008)
      REAL                DATA2(6912)
      CHARACTER*80     CINFILE
      INTEGER             IOS
      INTEGER             DATLEN
      INTEGER             NFIELDS

      CINFILE='/home/fr1800/frav/qrparm.orog.32.pp'
      NFIELDS=0

      OPEN(10,FILE=CINFILE,STATUS='OLD',ACCESS='SEQUENTIAL',
     &         FORM='UNFORMATTED',IOSTAT=IOS)

C Loop while there are no errors or no EOF

      DO WHILE (IOS.EQ.0)
        READ(10,IOSTAT=IOS) IHEAD,RHEAD

C Add in extra test when EOF has been reached

        IF (IOS.NE.0) GOTO 100

C Read in integer header values

        DO I=1,45
          WRITE(6,*) I,')  ',IHEAD(I)
        ENDDO

C Read in real header values

        DO I=1,19
          WRITE(6,*) I+45,')  ',RHEAD(I)
        ENDDO

C Set data array dimension according to header value 15
```

```
        DATLEN=IHEAD(15)
        IF (DATLEN.EQ.7008) THEN
          READ(10,IOSTAT=IOS) DATA1
        ELSEIF (DATLEN.EQ.6912) THEN
          READ(10,IOSTAT=IOS) DATA2
        ELSE
          WRITE(6,*) 'CANNOT READ IN DATA. ARRAY WRONGLY
      & DIMENSIONED'
          GOTO 200
        ENDIF

        NFIELDS=NFIELDS+1

C Add code to write out in your preferred format here

      ENDDO

100    CONTINUE
       WRITE(6,*) 'PP FILE CONTAINS ',NFIELDS,' FIELDS'

200    CONTINUE
       STOP
       END
```