```
                                type=shaper,content-out=p22

#   Unflatteners

chan        p2explode           prog=p2explode,type=shaper,content-in=p2
                                                                          280
chan        p22explode          prog=p2explode,type=shaper,content-in=p22


# Header Filters
chan    822tofax        prog=fcontrol,type=shaper,
                hdrin=822-us,hdrout=g3fax,
                outinfo="hdr2fax -from $(400sender) -to $(400recip) -subject $(ua-id) -table $(outtable) -ou
                cost=20,solo-proc=yes
chan        822touk             prog=fcontrol,type=shaper,
                                hdrin=822,                                290
                                hdrout=822-uk,
                                outinfo="rfc822norm -jnt -bigend"
chan        822tojnt            prog=fcontrol,type=shaper,
                                hdrin=822,hdrout=822-jnt,
                                outinfo="rfc822norm -jnt -bigend -jntsender $(822SENDER)"
chan        822tous             prog=fcontrol,type=shaper,hdrin=822,
                                hdrout=822-us,
                                outinfo="rfc822norm -822 -littleend"
chan        822top2             prog=RFCtoP2,type=shaper,hdrin=822-us,conv=1148
                                hdrout=p2,cost=10,                        300
                                show="RFC1148 RFC 822 -> P2"
chan        822top22            prog=RFCtoP2,type=shaper,hdrin=822-us,conv=1148
                                hdrout=p22,cost=10,
                                show="RFC1148 RFC 822 -> P22"
chan        p2to822             prog=P2toRFC,type=shaper,hdrout=822,conv=1148,
                                hdrin="ipn,p2",cost=10,
                                show="RFC1148 P2 -> RFC 822"
chan        p22to822            prog=P2toRFC,type=shaper,hdrout=822,conv=1148
                                hdrin="ipn,p22",cost=10,
                                show="RFC1148 P22 -> RFC 822"             310
chan        p22top2             prog=fcontrol,type=shaper,hdrout=p2,hdrin=p22,
                                show="P22 to P2 downgrade",
                                conv=conv,cost=10,outinfo="p2norm -downgrade"
#identity map
chan        p2top22             prog=fcontrol,type=shaper,hdrout=p22,hdrin=p2,
                                show="P2 to P22 upgrade",cost=1,
                                outinfo="/bin/cat"

# Body part filters
                                                                          320
chan    ia5-dk2ia5              prog=fcontrol,type=shaper,bptin=ia5-dk,
                                bptout=ia5,show="Danish ia5 to ia5",
                                conv=loss,outinfo="charset DK 38 UK 29"

chan    iso6937totxt            prog=fcontrol,type=shaper,bptin=iso6937,
                                bptout=ia5,conv=loss,cost=15,
                                outinfo="asn -inchar iso_6937-2-add -outchar irv -inasn motis-86-6937 -outasn ia
                                show="ISO6937 -> IA5"

chan        g3faxtotxt          prog=fcontrol,type=shaper,bptin=g3fax,    330
                                bptout=ia5,outinfo="removebp G3Fax",
                                show="Fax -> Text mapper",cost=100,conv=loss
chan    txttofax        prog=fcontrol,type=shaper,bptin=ia5,
```

```
                bptout=g3fax,
                        outinfo="ia52fax -table $(outtable)",outtable=fax,
                show="ia5 -> g3fax mapper"

chan    faxtops         prog=fcontrol,type=shaper,bptin=g3fax,
                        bptout=ps,outinfo="fax2ps",show="fax -> postscript"
                                                                        340
chan    mstops          prog=fcontrol,type=shaper,bptin=ms,
                        bptout=ps,outinfo="/usr/local/bin/psroff -ms -t",
                        show="ms macros -> postscript"

chan    mstoia5         prog=fcontrol,type=shaper,bptin=ms,
                        bptout=ia5,outinfo="/bin/cat",
                        show="MS->Text mapper",cost=100,conv=loss

chan    xxtotxt     prog=fcontrol,type=shaper,bptin=oid.0.9.2342.60200172.99.1.4,
                        bptout=ia5,outinfo="removebp oid.0.9.2342.60200172.99.1.4"
                        show="XXX -> Text mapper",cost=100,conv=loss

chan    voicetotxt      prog=fcontrol,type=shaper,bptin=voice,
                        bptout=ia5,outinfo="removebp Voice"
                        show="Voice -> Text mapper",cost=100,conv=loss

#  **-- Logging for the rest of the prog --**
#
authlog level=notice, size=400
operlog level=notice, size=400                                          360
normlog level=notice, size=400, sflags=zero

822flatten          normlog    file=filter
p2flatten           normlog    file=filter
822touk             normlog    file=filter
822tous             normlog    file=filter
dr2rfc              normlog    file=filter
fcontrol            normlog    file=filter
odif2txt            normlog    file=filter
p2explode           normlog    file=filter                              370
rfc822norm          normlog    file=filter
trash               normlog    file=filter

822top2             normlog    file=rfc987
p2to822             normlog    file=rfc987

list                normlog    file=lists

qmgr                normlog    file=qmgr
qmgr-load           normlog    file=qmgr                                380

smtp                normlog    file=smtp
smtpsrvr            normlog    file=smtp

greyout             normlog    file=greyout
gb-janet            normlog  file=greyin
submit              normlog    file=submit dflags=close

local               normlog    file=local
msg-clean           normlog    file=msg-clean                           390

x400in84            normlog    file=x400in84,level=pdus
```

```
x400in84        isodelog rtsaplevel file=rts,level=all
x400in84        isodelog ssaplevel file=ssap,level=all
x400out84       normlog  file=x400out84

x400in88        normlog  file=x400in88 level=pdus,level=trace
x400out88       normlog  file=x400out88 #level=all

p7server        normlog  file=p7server level=all                              400

xfax            normlog  file=fax #level=debug
xfaxd           normlog  file=fax #level=debug
G3Fax           normlog  file=fax
hdr2fax         normlog  file=fax
ia52fax         normlog  file=fax

#x400out88 isodelog      rtsaplevel file=rts-out      level=all
#x400out88 isodelog      acsaplevel file=acsap-out              level=all
#x400out88 isodelog      psap2level file=psap2-out              level=all   410
#x400out88 isodelog      ssaplevel file=ssap-out      level=all


#x400in88 isodelog       rtsaplevel file=proto-in               level=all
#x400in88 isodelog       acsaplevel file=proto-in               level=all
#x400in88 isodelog       psap2level file=proto-in               level=all

pptsapd         normlog  file=pptsapd,                dlevel=notice
dirlist         normlog  file=dirlist
mslocal         normlog file=mslocal                                          420
xprobe          normlog file=probe level=trace,level=pdus level=all
p7server        isode ns_enable off
xua             isode ns_enable off
libmsattr               isode ns_enable off
#xckadr normlog level=all file=ckadr
```

Figure A.1: An Example Tailor File

Appendix B

Example of the PP Start Script

What follows is an example of a script which is run to start the PP mail system.

```
#! /bin/sh

# SMTP running under inetd
#cd /usr/pp/pp/lib/chans
#if [ -f smtpd -a -f smtpsrvr ]; then
#           (./smtpd `pwd`/smtpsrvr smtp > /dev/null 2>&1 ) &
#           echo -n ' smtp'
#fi

cd /usr/pp/pp/lib                                                    10
su pp <<EOF
echo -n ' pp.startaspp :  '
if [ -f pptsapd ]; then
           echo -n ' pptsapd '
           pptsapd > /dev/null 2>&1
fi
if [ -f qmgr ]; then
           echo -n ' qmgr '
           qmgr > /dev/null 2>&1
fi                                                                   20
echo '.'
EOF
```

Figure B.1: An Example Start Script

Appendix C

Example of the PP Script Run Nightly

What follows is an example of a script which is regularly run to keep the PP mail system up to date.

```
#! /bin/sh

PATH=/crg/pp/bin:/crg/pp/cmds/tools:/usr/local/bin:/usr/ucb:/bin:/usr/bin:/usr/5bin:
export PATH
#
# Shell script run nightly to collect up things etc.
#
set -x
cd /crg/pp
L=logs                                                                    10
T=tables
STATDIR=/crg/pp/logs/statistics

exec 1> $L/pp-nightly.log 2>&1

freespace . 2000 || { echo NO space left on device; exit 1; }

#Save the stats files
if [ -f $L/stat ]
then                                                                      20
        DATE=`date +%h-%d.%T`
        [ -d $L/tmp ] || mkdir $L/tmp
        mv $L/stat $L/tmp/stat.$DATE
fi

# And then once a week...
if [ `date +%w` -eq 0 ]
then
        YDATE=`date "+%h-%d-%y`
        [ -d $STATDIR ] || mkdir $STATDIR                                 30
        if pstat $L/tmp/* /dev/null | bin/stat.awk > $STATDIR/stats.$YDATE
        then
                echo 'Sucessful - removing old files'
                rm $L/tmp/*
        fi
fi
```

```
# This should be first building command, as it updates the DERFIL2 file
echo 'building the PP tables'
(cd tables; make install)                                                    40

# update the niftp stuff
echo 'building the niftp database'
(cd niftp; make )

echo UPDATE complete
```

Figure C.1: An Example Nightly Script

Appendix D

Obtaining X.400 Tables

The following tables are necessary for correct X.400 operation. These tables should be obtained from UCL. The tables are:

`or:` The O/R Table.

`rfc2or:` The RFC 822 to O/R mapping table

`or2rfc:` The O/R to RFC 822 mapping table.

`rfc1148gate:` The RFC 1148 known gateways file.

These tables can be fetched in a number of ways.

FTP: If you have access to Internet FTP then use anonymous FTP to cs.ucl.ac.uk [128.16.5.31] to retrieve the files from the directory *pp-tables/*. For example:

```
% ftp cs.ucl.ac.uk
Connected to bells.cs.ucl.ac.uk.
220 bells.cs.ucl.ac.uk FTP server (SunOS 4.0) ready.
Name (cs.ucl.ac.uk:pp): anonymous
331 Guest login ok, send ident as password.
Password:
230 Guest login ok, access restrictions apply.
ftp> get pp-tables/or or
...
ftp> get pp-tables/or2rfc or2rfc
...
ftp> close
%
```

FTAM: If you have access to FTAM on Janet, PSS or the Internet, you may obtain the tables from UCL using one of the following:

- Janet DTE 00000511160013 with TSEL 259 (ascii-encoded)
- PSS DTE 23421920030013 with TSEL 259 (ascii-encoded)
- Internet address cs.ucl.ac.uk ([128.16.5.31]) port 102 with TSEL 259 (ascii-encoded)

Use the "anon" user-identity and retrieve the files in the *pp-tables/* directory. For example:

```
% ftam ucl
user (ucl:pp): anon
ucl... connected
ANONymous user permitted, access restrictions apply
ucl> get pp-tables/or /tmp/or
ftam: unstructured text file transfer
ftam: 5792 bytes received in 5.88 seconds (0.96 Kbytes/s)
...
ucl> close
%
```

NIFTP: If you run NIFTP over the public X.25 or over Janet, and are registered in the NRS at salford, you can use NIFTP with username "guest" and your name as password, to access UK.AC.UCL.CS to retrieve the files found under `<pp-tables>`. For example:

```
% cpf '<pp-tables>or'@uk.ac.ucl.cs or
%
```

MAIL: The files may be obtained by mail by sending a suitable query to the address `info-server@cs.ucl.ac.uk`. This query should consist of a message body with the lines:

```
Request: pp-tables
Topic: or
Topic: rfc2or
Topic: or2rfc
Topic: rfc1148gate
```

E.g.:

```
% mail info-sver@cs.ucl.ac.uk <<EOF
Request: pp-tables
Topic: or
Topic: rfc2or
Topic: or2rfc
Topic: rfc1148gate
EOF
%
```

Appendix E

Testing PP — the MTS User Interface

A special user interface known as *mu* is provided as a general tool to help test *submit* and consequent operations in PP. Setting defaults in the *mu.defaults* file allows a minimum number of message-specific parameters to be provided. It may be used to resubmit a modified text-encoded file taken from the queue.

E.1 Overview

The message submission tool MU takes input from the command line, from a defaults file *mu.defaults* if present in the working directory, and otherwise from the default command directory `cmddfldir` and then any other files specified with `-file` flags. A full summary of the flags to MU is given in Table E.1 on page 152. Any command line option may be placed on a line within an input file. MU calls its command line parser recursively when it encounters a line starting with `'-'` (flagchar). Any line starting with `'#'` (commentchar) is ignored. Otherwise, all lines are offered to the various *submit* input routines in turn for incorporation into PP's internal message structures (*prm_vars*, *Qstruct* and *addr*). Any line which returns an error code (`rp_isbad`) from all these routines is assumed to be in error.

The input lines which specify assignments as specified in queue (5) may be received in any sequence, and inputs may override earlier entries. Any value may also be assigned on the (original) command line with `-value v1 v2` ... up to the next flag.

If input files are nested, input lines from the nested file are processed before the remainder of the file containing the `-file` specifier. The default file is parsed first (*mu.defaults* unless overridden with a `-default` flag on the command line), then any `-file` specified files from the command line, finally `-value` specifications from the command line. Body part files may be specified with `-body` commands, either on the command line or embedded as with other commands.

MU uses PP's standard logging: additional logging messages generated

by –0 –1 –2 –3 flags are output to a channel (`muout`) which defaults to `stdout` unless flag `-logfile` is specified. Note that output may be rerouted in the course of processing input files. The flag –5 allows these messages to be appended to the nominated logfile, rather than replacing it.

In addition to the llog_trace and llog_debug calls associated with flags `-0` and `-1` shown below, there are some other llog_fatal and llog_exceptions calls. Note that functions `list_args` and `show_args` (in *argh.c*) do not generate llog events. MU accepts a number of debug flags `-0`, `-1` etc.; currently these are used as follows:

flag	function
-0	Log to *muout* progress messages, and pplog(llog_trace ...).
-1	Log to *muout* command files as they are processed, and pplog(llog_debug ...).
-2	List to *muout* body part files as they are io_tdata'ed.
-3	List to *muout* structures prior to *submit* io_ calls.
-4	Abort prior to *submit* io_ calls.
-5	Append to logfile rather than overwrite.

MU accepts `-body` part file names with any arbitrarily long pathname. The full path is used:

1. To open the file and check its length. Two fatal errors are detected :
   a. `unable to read body part %s`.
   b. `body part %s is null file`.
2. To open the file in order to pass its contents to *submit* with a series of *io_tdata(buf,length)* calls. Again, failure to open is reported:
   `unable to open body part for reading %s`.
3. To pass a name for *submit* to use in the queue. *submit* generates a message directory to match the supplied pathname. Thus:
   `-body 4.ipm/hdr.822uk` — generates :
   usr/spool/pp/queues/msg.annnnnn/base/4.ipm/hdr.822uk

If a reserved character "*" (filemarkchar) is embedded in the `-body` pathname, MU strips all characters up to and including the "*' from the pathname before passing to *submit*. This allows MU to distinguish between the path used to find the file(s) and that required within the message. Thus:

```
-body nest1/nest2/*3.ipm/1.ia5
```

generates:

```
/usr/spool/pp/queues/msg.annnnnn/base/3.ipm/i.ia5
```