```
void    adios (what, fmt)
char   *what,
       *fmt;
{
    adios (what, fmt);
}
#endif                                                          70


#ifndef lint
void    advise (va_alist)
va_dcl
{
    int     code;
    va_list ap;

    va_start (ap);                                              80

    code = va_arg (ap, int);

    _ll_log (pgm_log, code, ap);

    va_end (ap);
}
#else
/* VARARGS3 */
                                                                90
void    advise (code, what, fmt)
char   *what,
       *fmt;
int     code;
{
    advise (code, what, fmt);
}
#endif
```

10.1.4 An Example

An example of a responder written using this boilerplate is shown in Section 11.4 on page 184.

Chapter 11

A Complete Example

Now, let's consider a complete example. Our application is modeled after the example given in [Coo82]. In short, we wish to implement a remote lookup facility of the UNIX */etc/passwd* database.

11.1 Service Definitions

We begin by defining the mundane naming and addressing information for the service. Using the "standard" approach described in Section 3.1, we select:

abstract syntax: defined in the *isobjects*(5) file as:

```
"local service pci"      1.17.2.n.1
```

If we select `n` as the lowest unassigned number in the `1.17.2` subtree, e.g., 7, we might have:

```
"isode passwd lookup pci"      1.17.2.7.1
```

application context name: defined in the *isobjects*(5) file as:

```
"local service"          1.17.2.n.2
```

Similarly, for a value of 7 for `n`, we have:

```
"isode passwd lookup"          1.17.2.7.2
```

Directory entry: if we select `p` as the lowest unassigned TSAP ID between `1024` and `2047` inclusive, e.g., `1040`, and decide to name the service as `passwdstore`, we have:

```
objectClass= top & applicationEntity & quipuObject & iSODEApplicationE
cn= passwdstore
presentationAddress= #1040/Internet=mydomainame
supportedApplicationContext= isode passwd lookup
acl=
execVector= ros.lookup
```

11.2 Remote Operations Module

The remote operations module for the service is placed in *lookup.ry*. This file shown in Figure 11.1 starting on page 173.

The *Makefile* for the service is shown in Figure 11.2 starting on page 177.

```
PasswordLookup DEFINITIONS ::=

BEGIN

-- operations

                        -- given a user name, return a Passwd type
lookupUser OPERATION
        ARGUMENT        UserName
        RESULT          Passwd                                      10
        ERRORS          { noSuchUser, congested }
        ::=             0

                        -- given a user ID, return a Passwd type
lookupUID OPERATION
        ARGUMENT        UserID
        RESULT          Passwd
        ERRORS          { noSuchUser, congested }
        ::=             1
                                                                    20
-- errors

                        -- no matching user in the database
noSuchUser ERROR
        ::=             0

                        -- congestion at responder
congested ERROR
        ::=             1
                                                                    30
-- types

                        -- similar to an entry in <pwd.h>
Passwd ::=
   [APPLICATION 1]
        IMPLICIT SEQUENCE {
           name[0]
                IMPLICIT UserName,
```

```
        passwd[1]                                                    40
                IMPLICIT IA5String
                OPTIONAL,

        uid[2]
                IMPLICIT UserID,

        gid[3]
                IMPLICIT GroupID,

        quota[4]                                                     50
                IMPLICIT INTEGER
                DEFAULT 0,

        comment[5]
                IMPLICIT IA5String
                OPTIONAL,

        gecos[6]
                IMPLICIT IA5String
                OPTIONAL,                                            60

        dir[7]
                IMPLICIT IA5String
                OPTIONAL,

        shell[8]
                IMPLICIT IA5String
                OPTIONAL
    }
                                                                     70

UserName ::=
   [APPLICATION 2]
        IMPLICIT GraphicString

UserID ::=
   [APPLICATION 3]
        IMPLICIT INTEGER
```

```
GroupID ::=
   [APPLICATION 4]                                          80
        IMPLICIT INTEGER

END
```

Figure 11.1: Remote Operations Module for the Password Lookup Service

```
#   Instructions to Make, for compilation of ROS-based password lookup service

BINDIR =        /usr/new/
SBINDIR =       /usr/etc/
ETCDIR =        /usr/etc/

CC      =       /bin/cc
CFLAGS =        -g    $(OPTIONS)
LINT    =       lint
LFLAGS =        -bhuz $(OPTIONS)                                        10
LLIBS   =       -lisode
LD      =       ld
LDCC    =       /bin/cc
LDFLAGS =       -g
LIBES   =       -lisode

HFILES =        ryinitiator.h ryresponder.h
CFILES =        lookupd.c lookup.c ryinitiator.c ryresponder.c
RYFILES =       lookup.ry
                                                                        20

# Here it is...

all:            lookupd lookup
inst-all:       inst-lookupd inst-lookup # manuals
install:        inst-all clean
lint:           l-lookupd l-lookup


# lookupd                                                               30

inst-lookupd:   $(SBINDIR)ros.lookup

$(SBINDIR)ros.lookup:    lookupd
                -cp $@ zros.lookup
                -rm -f $@
                cp lookupd $@
                -@ls -gls $@
```

```
		-@echo ""
lookupd:	lookupd.o PasswordLookup-Rops.o \
			ryresponder.o PasswordLookup_tables.o
		$(LDCC) $(LDFLAGS) -o $@ lookupd.o \
			PasswordLookup-Rops.o \
			ryresponder.o
			PasswordLookup_tables.o \
			$(LIBES)

l-lookupd:	PasswordLookup-ops.c PasswordLookup_tables.c true
		$(LINT) $(LFLAGS) -DPERFORMER lookupd.c \
			PasswordLookup-ops.c ryresponder.c \
			PasswordLookup_tables.c $(LLIBS)

lookupd.o:	ryresponder.h PasswordLookup-ops.h \
			PasswordLookup-types.h

PasswordLookup-Rops.o: PasswordLookup-ops.c PasswordLookup-ops.h
		$(CC) $(CFLAGS) -DPERFORMER -c PasswordLookup-ops.c
		mv PasswordLookup-ops.o $@

ryresponder.o:	ryresponder.h

PasswordLookup_tables.o: PasswordLookup_tables.c PasswordLookup-types.h

PasswordLookup_tables.c PasswordLookup-types.h:  PasswordLookup-asn.py
		pepsy -A -f -h -m PasswordLookup-asn.py

PasswordLookup-asn.py:  lookup.ry
		rosy -m -pepsy -o $@ lookup.ry
PasswordLookup-ops.c:   lookup.ry
PasswordLookup-ops.h:   lookup.ry
PasswordLookup-stubs.c: lookup.ry


# lookup

inst-lookup:	$(BINDIR)lookup
```

```
$(BINDIR)lookup:        lookup
                -cp $@ zlookup                                          80
                -rm -f $@
                cp lookup $@
                -@ls -gls $@
                -@echo ""

lookup:         lookup.o PasswordLookup-Iops.o \
                        ryinitiator.o PasswordLookup_tables.o
                $(LDCC) $(LDFLAGS) -o $@ lookup.o \
                        PasswordLookup-Iops.o \
                        ryinitiator.o \                                 90
                        PasswordLookup_tables.o \
                        $(LIBES)

l-lookup:       PasswordLookup-ops.c PasswordLookup_tables.c true
                $(LINT) $(LFLAGS) -DINVOKER lookup.c \
                        PasswordLookup-ops.c ryinitiator.c \
                        PasswordLookup-stubs.c \
                        PasswordLookup_tables.c $(LLIBS)

lookup.o:       ryinitiator.h PasswordLookup-ops.h PasswordLookup-types.h

PasswordLookup-Iops.o:  PasswordLookup-ops.c PasswordLookup-ops.h
                $(CC) $(CFLAGS) -DINVOKER -c PasswordLookup-ops.c
                mv PasswordLookup-ops.o $@

ryinitiator.o:  ryinitiator.h


# manual pages
                                                                        110
MANUALS =       lookupd.8c lookup.1c

manuals:;       install -c -m 0444 lookupd.8c \
                        /usr/man/man8/lookupd.8c
                install -c -m 0444 lookup.1c /usr/man/mann/lookup.n
```