

CMSC 818S

DataCutter tutorial

Christian Hansen
November 12, 2002

CMSC 818S - Alan Sussman

1

```
// Filter specification
class Echo: public DC_Filter_Base_t {
public:
    DC_RTN_t init(initarg_t &initarg);
    DC_RTN_t process(arg_t &arg);
    DC_RTN_t finalize(void);
private:
    DC_Buffer_t buf;
    DC_Buffer_t* pbuf;
};
```

```
// Obtain the work definition
```

```
DC_RTN_t Echo::process(arg_t &arg) {
    int len;
    char str[STRLEN];

    char* msg = arg.pwork->buf.getPtr();
    cout << "work def: << msg << endl;
```

```
// Read from a stream
```

```
while ((pbuf = arg.ins[0].read())) {
    pbuf->Extract(&len);
    pbuf->Extract(str, len);
    cout << str;
    pbuf->consume();
}
```

```
// Write to a stream
```

```
buf.New(BUFLEN);
buf.Append(len);
buf.Append(str, len);
if (arg.outs[0].write(&buf) != DC_ERR_OK) {
    cerr << arg.sbFilterName << ": failed write" << endl;
    return DC_RTN_ABRT;
}

return DC_RTN_OK;
}
```

```
// Filter creation
```

```
DC_Filter_Base_t *filter_factory(char *sbFilterName) {
    if (strcmp(sbFilterName, "Read") == 0) {
        return new Read;
    } else if (strcmp(sbFilterName, "Echo") == 0) {
        return new Echo;
    } else {
        cerr << "unknown filter: " << sbFilterName << endl;
    }
    return NULL;
}
```

```

// Application startup

int main(int argc, char* argv[]) {
    DC_FilterService_t dc;
    if (dc.init("EchoApp", &filter_factory, &argc, &argv)) {
        exit(1);
    }

    if (dc.isRemoteProcess()) {
        dc.RemoteProcess();
        return 0;
    }
}

```

```

// Console – Layout, Placement, Work

```

```

DC_FilterLayout_t lyt;
lyt.Add("Reader", NULL, "r2e");
lyt.Add("Echo", "r2e", "e2c");
lyt.Add("<console>", "e2c");

DC_Placement_t plc;
plc.Add("Read", "redleader");
plc.Add("Echo", "<one_per_node>");

DC_Work_t wrk(FILEBLOCK);
wrk.buf.Empty();
wrk.buf.Append(strlen(def)+1);
wrk.buf.Append(str, strlen(def)+1);

```

```

// Console – Instance startup

```

```

int sts;

DC_FilterInstance_t* inst;
if((sts = dc.NewFilterInstance(lyt, wrk, plc, inst)) !=
    DC_ERR_OK) {
    cerr << DC_strerror(sts) << endl;
    return 1;
}

```

```

// Console – Work assignment

```

```

int wh;
if ((wh = inst->AppendWork(wrk) < 0) {
    cerr << DC_strerror(wh) << endl;
    return 0;
}

DC_Buffer_t* pbuf;
while ((pbuf = inst->arg.ins[0].read())) {
    pbuf->consume();
}

```

```

// Console – Work and instance shutdown

```

```

sts = dc.WaitWork(wh);
cerr << DC_strerror(sts) << endl;
if (sts == DC_ERR_InstanceFail) {
    exit(1);
}

if ((sts = dc.StopFilterInstance(inst)) != DC_ERR_OK) {
    cerr << DC_strerror(sts) << endl;
}
return 0;
}

```

Compilation

- Include DataCutter.h
- Link to libDC.a
- Makefile:

```

ifeq $(OSTYPE),linux)
    DEFS = -DHAVE_CONFIG_H -DDEBUGGER_USE_GDB -DLINUX
    LIBS = -L../dc/lib -lDC -lmsl -lpthread
endif

ifeq $(OSTYPE),solaris)
    DEFS = -DHAVE_CONFIG_H -DDEBUGGER_USE_GDB -DSUNOSS
    LIBS = -L../dc/lib -lDC -lmsl -lsocket -lthread -lposix4
endif

```

Environment

- Add the run-time binaries to your path
 - /redleader/chansen/work/DataCutter-2.1/bin
- Set `DATA CUTTER_D IRDHOST` and `DATA CUTTER_D IRDPORT`
- Generate authentication keys for ssh
 - `man ssh-keygen`

Starting the Run-Time System

- Start `dird`
 - `boot_util dird start`
- Start `appds`
 - `boot_util appd start rogue01 rogue02 ...`
- Trouble?
 - Can you ssh into the remote host?
 - Can you start `dird/appd` manually?

Starting your application

- Put your executable in a directory common to all hosts
 - Just use your home directory on redleader
- Run your application and watch the output
 - Filter output will appear in small grey terminal windows created by `boot_util`