HMI Team Workshop: Local Helioseismology Pipelines, March 2007

A Quick Look at DRMS

Art Amezcua
Stanford University
arta@sun.stanford.edu

What is DRMS?

- Data Record Management System
- Way to organize and store vast amounts of data
- Implementation
  - Wrapper around PostgreSQL Database
  - Uses SUMS for data file storage and retrieval
- The interface to JSOC data (via a C API)

What is SUMS?

- Storage Unit Management System
- Manages JSOC's disk and tape resources, shuttling data between the two
- DRMS is SUMS' only client
- Accepts requests from DRMS that specify how to store data

How do I use DRMS?

- Client-Server Architecture – To use DRMS, a DRMS client-server session must be started. Modules (the clients) make requests to the server. Changes saved on session shut down.
- Modules – Stand-alone executables linked to jsoc_main, the DRMS driver.
- Session Providers – These start a DRMS server, launch scripts/modules/executables that use the DRMS API, shut down server. Examples:
  - drms_run
  - PUI (Pipeline User Interface)
  - Rick's web interface.

Basic DRMS Architecture

Dataseries – A set of data records. Has a primary index.

Data record – Keyword (metadata) plus zero or more data segments and links.

Data segment - Individual data object, such as a file, plus bookkeeping .

PostgreSQL representation of data series with primary index obs_date:

recnum	obs_date	fmoon	seg0	seg1
2	02/16/2007	no	/SUM3/D707538/S0000/lr.jpg	/SUM3/D707538/S0000/hr.jpg
3	03/03/2007	yes	/SUM3/D707549/S0000/lr.jpg	/SUM3/D707549/S0000/hr.jpg
5	02/16/2007	no		/SUM3/D707558/S0000/hr.jpg

APIs to Access Records

```
DRMS_RecordSet_t *drms_open_records(DRMS_Env_t *env,
                                    char *recordsetname,
                                    int *status);
```

- su_arta.TestSeries
- su_arta.TestSeries[:#5]
- su_arta.TestSeries[OBS_DATE=2009.11.21_05:00_UTC]

```
DRMS_RecordSet_t *drms_clone_records(DRMS_RecordSet_t *recset,
                                     DRMS_RecLifetime_t ltime,
                                     DRMS_CloneAction_t mode,
                                     int *status);

DRMS_RecordSet_t *drms_create_records(DRMS_Env_t *env, int n,
                                      char *series,
                                      DRMS_RecLifetime_t ltime,
                                      int *status);

int drms_close_records(DRMS_RecordSet_t *rs, int action);
```

Accessing Records Snippet (1)

```
DRMS_RecordSet_t *recSet = drms_open_records(drms_env, recSetStr, &status);
DRMS_RecordSet_t *recSetClone = NULL;

if (status == DRMS_SUCCESS) {
   recSetClone = drms_clone_records(recSet, DRMS_PERMANENT,
                                    DRMS_SHARE_SEGMENTS, &status);
   if (status == DRMS_SUCCESS) {
      /* Do something with recSetClone here, like update a keyword. */
   }
}

if (recSet != NULL) {
  drms_close_records(recSet, DRMS_FREE_RECORD);
}
if (recSetClone != NULL) {
   drms_close_records(recSetClone, DRMS_INSERT_RECORD);
}
```

Accessing Records Snippet (2)

```
DRMS_RecordSet_t *recSet = drms_create_records(drms_env, 1,
                                               seriesName, DRMS_PERMANENT,
                                               &status);

if (status == DRMS_SUCCESS) {
   DRMS_Record_t *rec = recSet->records[0];

   /* Set a keyword value */
   drms_setkey(rec, keyname, keytype, &value);
   drms_close_records(recSet, DRMS_INSERT_RECORD);
}
```

APIs to Access Keywords

Obtain a Keyword

- `DRMS_Keyword_t *drms_keyword_lookup(DRMS_Record_t *rec, const char *key, int followlink);`

'Get' ('Set') Keyword Values APIs

- `char drms_getkey_char(DRMS_Record_t *rec, const char *key, int *status);`
- `short drms_getkey_short(DRMS_Record_t *rec, const char *key, int *status);`
- `int drms_getkey_int(DRMS_Record_t *rec, const char *key, int *status);`
- `long long drms_getkey_longlong(DRMS_Record_t *rec, const char *key, int *status);`
- `float drms_getkey_float(DRMS_Record_t *rec, const char *key, int *status);`
- `double drms_getkey_double(DRMS_Record_t *rec, const char *key, int *status);`
- `char *drms_getkey_string(DRMS_Record_t *rec, const char *key, int *status);`
- `DRMS_Type_Value_t drms_getkey(DRMS_Record_t *rec, const char *key, DRMS_Type_t *type, int *status);`

Accessing Keywords Snippet

```
DRMS_Keyword_t *key = drms_keyword_lookup(record, keyName, 1);
char *str = NULL;
DRMS_Type_Value_t value;

if (key != NULL) {
   str = drms_getkey_string(record, keyName, &status);
   if (status == DRMS_SUCCESS) {
      snprintf(buf, sizeof(buf), "%s_ch", str);
      value.string_val = strdup(buf);
      status = drms_setkey(record, diffKeyName, DRMS_TYPE_STRING,
                           &value);
   }

   if (str) {
      free(str);
   }
}
```

APIs to Access Segments

- ```
  DRMS_Segment_t *drms_segment_lookup(DRMS_Record_t *rec,
                                      const char *segname)
  ```
- ```
  DRMS_Array_t *drms_segment_read(DRMS_Segment_t *seg,
                                  DRMS_Type_t type,
                                  int *status)
  ```
- ```
  int drms_segment_write(DRMS_Segment_t *seg,
                                  DRMS_Array_t *arr,
                                  int autoscale);
  ```
- ```
  static inline
  long long drms_array_count(DRMS_Array_t *arr);
  ```
- ```
  DRMS_Array_t *drms_array_create(DRMS_Type_t type,
                                  int naxis, int *axis,
                                  void *data, int *status);
  ```
- ```
  void drms_free_array(DRMS_Array_t *src);
  ```

Accessing Segments Snippet

```
DRMS_Segment_t *segIn = drms_segment_lookup(rec, segNameIn);
DRMS_Segment_t *segOut = drms_segment_lookup(rec, segNameOut);

if (segIn != NULL && segOut != NULL) {
   DRMS_Array_t *dArr = drms_segment_read(segIn, DRMS_TYPE_FLOAT, &status);
   if (status == DRMS_SUCCESS) {
      int nElements = drms_array_count(dArr);
      float *data = (float *)dArr->data;
      int idx = 0;

      for (; idx < nElements; idx++)
         newData[idx] = data[idx] * 2.7;

      dArrNew = drms_array_create(DRMS_TYPE_FLOAT, ndims, dimarr, newData,
                                  &status);

      if (status == DRMS_SUCCESS) {
         drms_segment_write(segOut, dArrNew, 0);
         drms_free_array(dArrNew);
      }
      drms_free_array(dArr);
   }
}
```