PAUL SCHERRER INSTITUT

WIR SCHAFFEN WISSEN – HEUTE FÜR MORGEN

Mark Könnecke

MongoDB Handson
December, 7, 2016

Disclaimer

- This has very little to do with ESS and the joint projects between PSI and ESS!
- This is no suggestion from PSI for the joint ESS-PSI projects
- Michael just asked me to present experiences from yet another project

So, what did we do? And why?

- In collaboration with ANSTO, PSI revised the logging system of our in house DAQ software SICS
- Merge three different logging facilities in SICS to one!
  - Lots of code was changed and cleaned up
- Output to ASCII log files for all us old women and men
- Provided an adapter to listen in to the log, a callback
- This callback was used to implement logging into a MongoDB

Why MongoDB?

- There is no enterprise logging service at PSI
- As I was not sure that logging to a database is for us, I wanted something under my control
- I wanted flexibility regarding the DB schema
- I wanted to play with a non NoSQL DB anyway
- MongoDB was quicker to setup then MySQL

What do we log?

- timestamp, severity, subsystem, message
  - Implicit: instrument
- severity:
  - fatal, error, warn, info, verbose, debug
- subsystems:
  - sys, com, asqio, io, dev, par, notify, history
- A further subdivision beyond subsystem, like source could be useful
- timestamps: ISO8601
- Logging can be configured online:
  - global log level
  - enable full logging for sub systems

C-code: Initialization and closing

```
    mongoc_init();
    client = mongoc_client_new (mongoURL);
    collection = mongoc_client_get_collection (client, inst, "log");

============================================================================
static void MongoClose(void *data){
  RemoveLogCallback(MongoCallback);
  if(collection != NULL){
    mongoc_collection_destroy (collection);
    mongoc_client_destroy (client);
    mongoc_cleanup ();
  }
  client = NULL;
  collection = NULL;
}
```

C-Code: Entering log data

```
 tVal = DoubleTime();
doc = bson_new ();
bson_oid_init (&oid, NULL);
BSON_APPEND_OID (doc, "_id", &oid);
BSON_APPEND_DOUBLE(doc,"timestamp",tVal);
BSON_APPEND_INT32(doc,"severity",severity);
BSON_APPEND_UTF8(doc,"timetext",timeStamp);
BSON_APPEND_UTF8(doc,"sub",subsystem);
BSON_APPEND_UTF8(doc,“message",message);
mongoc_collection_insert (collection, MONGOC_INSERT_NONE, doc, NULL, &error)) ;
bson_destroy (doc);
```

Querying the MongoDB Log

- Using C-code:
  - For internally within SICS
  - As external CLI program
- In python
  - Try out the python MongoDB API
  - Get some statistics

C-code: Querying

```
 snprintf(jsonQuery, sizeof(jsonQuery),"{ \"timestamp\" : {\"$gt\": %ld, \"$lt\": %ld}, 
\"severity\": {\"$lte\": %d}“, from,to,severity);

  if(sub != NULL){

    snprintf(subQuery,sizeof(subQuery),", \"sub\" : \"%s\"", sub);

    strncat(jsonQuery,subQuery,sizeof(jsonQuery) -strlen(jsonQuery)-2);

  }

…..

  collection = mongoc_client_get_collection(client,instLocal,"log");

  query = bson_new_from_json((const uint8_t *)jsonQuery,strlen(jsonQuery), &err);

  cursor = mongoc_collection_find (collection, MONGOC_QUERY_NONE, 0, 0, 0, query, 
NULL, NULL);

  while (mongoc_cursor_next (cursor, &doc)) {

    ResultPrint(doc,userData);

  }
```

C-code: Printing Query Results

```
static void  ResultPrint(const bson_t *doc, void *userData) {
  bson_iter_init(&iter,doc);
  bson_iter_find(&iter,"timetext");
  timeText = bson_iter_utf8(&iter,&length);
  bson_iter_init(&iter,doc);
  bson_iter_find(&iter,"severity");
  severity = bson_iter_int32(&iter);
  formatSeverity(severity,sevBuf,sizeof(sevBuf));
  bson_iter_find(&iter,"sub");
  sub = bson_iter_utf8(&iter,&length);
  bson_iter_find(&iter,"message");
  message = bson_iter_utf8(&iter,&length);

  fprintf(stdout,"%s    %s   %s   %s\n",timeText,sevBuf, sub, message);


}
```

Python API: errorsummary

```
import pymongo
from pymongo import MongoClient
from datetime import timedelta
mongodb = MongoClient('mongodb://XXXXXXX')
dblist = ['amor', 'boa', 'dmc', 'eiger', 'focus', 'hrpt', ,morpheus‘,….]
for inst in dblist:
    db = mongodb[inst]
    ct = db.log.find({"timestamp" : {"$gt": time.mktime(fromtime.timetuple()),
                         "$lt" : time.mktime(totime.timetuple())},
                          "severity": {"$lte": 2} }).count()
    print(inst + ',' + str(ct))
```

Errorsummary Output

```
Error counts between 2016-12-01 00:00:00 to 2016-12-02 00:00:00
amor,0
boa,63
dmc,35
eiger,21
focus,75
hrpt,3305
morpheus,495
narziss,56
orion,32
poldi,437
rita2,0
sans,2887
sans2,138
tasp,114
zebra,89
```

Python-API: Error analysis

```
import pymongo
from pymongo import MongoClient
from datetime import timedelta
inst = sys.argv[1]
mongodb = MongoClient('mongodb://XXXXX')
messagedict = {}
db = mongodb[inst]
for entry in db.log.find({"timestamp" : {"$gt": time.mktime(fromtime.timetuple()),
                "$lt" : time.mktime(totime.timetuple())}, "severity": {"$lte": 2} }):
                mes = entry['message']
                if messagedict.has_key(mes):
                  messagedict[mes] = messagedict[mes] + 1
                else:
                  messagedict[mes] = 1
for mes in sorted(messagedict, key=messagedict.get):
   print('%8d %s' %(messagedict[mes], mes ))
```

Linecounts

- Logging in C: 204 lines
- Querying in C: 281 lines
- Python errorsummary: 24 lines
- Python errorstatistics: 35 lines
- About a week worth of work
  - Most time spent learning the API’s

SINQ MongoDB Setup

- This was meant as a test
- HP Z400 from 2010, 4GB memory, 3TB disk
  - Holding up well for 16 instruments
  - Memory is tight for MongoDB
- I instinctively partioned the database according to instruments
- At first querying was slow
  - Setting an index on the timestamp sorted that

Some Statistics

- Number of messages/instrument/day
  - Sit and count: 10 - 15 K
  - Scanning: up to 70 K
  - Full output, all I/O: up to 7 Million messages
    - This then takes ~ a minute to query
  - ~123 GB for 6 months of SINQ operation
  - But we had an unplanned shutdown since june
- I am still tuning what goes into the log

Summary and Outlook

- No problems with MongoDB so far
- Securing a MongoDB instance is difficult
- I have not tried load balancing etc.
- Having a queryable log helped a lot in debugging problems
  - I regret not having implemented something like this earlier
- Will move to PSI provided MongoDB server in 2017