RFC: Refactoring Library for HDF5 Tools

Peter Cao
Jonathan Kim
Albert Cheng
Allen Byrne

This RFC discusses how to restructure library for HDF5 tools. Currently there is no clear structure for a tool developer to follow as implementing new functions either for updating current tools or developing a new tool. This causes repeated code and functions both in the tool library and tools. It significantly downgrades the quality of code and also reduces productivity for a developer not to be able to easily search and reuse functions already implemented before.

1 Introduction

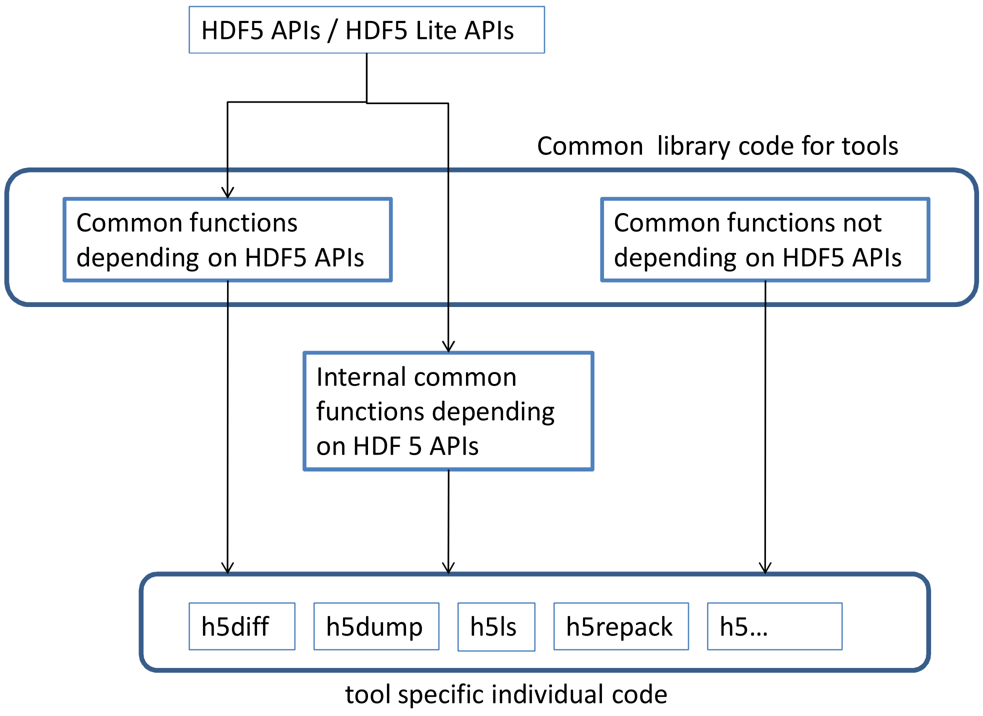
This proposal explains a layout structure for the tool library and naming conventions. This suggestion will set a structure so a tool developer can more effectively review the existing functions to reuse. If no function is suitable to reuse, then the developer can implement new functions and should know where to locate them either to be reused in the future or just to use for a specific tool.

2 Motivation and Goal

- Improve code quality and remove repeated code or functions.
- Internal to THG: effective coding, improve productivity and easy maintenance.
- External to users: any user can use the common functions for their own program as necessary. Also encourage for user to improve THG tools.

3 Structural layout design

Divide code layout to ‘common code’ and ‘tool specific code’. And also divide ‘common code’ to ‘functions depending on HDF5 APIs’, ‘Internal functions depending on HDF5 APIs’ and ‘functions not depending on HDF5 APIs’. See below for more details.


HDF5 APIs / HDF5 Lite APIs
Common library code for tools
Common functions depending on HDF5 APIs
Common functions not depending on HDF5 APIs
Internal common functions depending on HDF 5 APIs
h5diff
h5dump
h5ls
h5repack
h5…
tool specific individual code

3.1 Common functions depending on HDF5 API

- A function can be used by other tools.
- A function can be integrated by users for utilizing tool's feature in their own program.
- Feature details
  - Traversing
  - Comparing
  - Searching
  - Fetching
  - Importing/Exporting
  - Copying
  - Objects manipulating (create, remove, rename)
  - Status checking
  - Debugging
  - <<Continue adding functions as work progresses>>

3.2 Common functions not depending on HDF5 API

- Functions to be used by any tool. (possibly by users too)
- Feature details
  - Print out the result various way (OUTPUT)
  - Command parameter passing options scheme (INPUT from user)
  - Error handling
  - Debugging printouts
  - Memory handling
  - <<continue adding functions as work progresses>>

3.3 Internal common functions depending on HDF5 APIs

- Any functions that can be used by other tools can be put here.
- When a tool developer implements a general function, he or she will make a decision whether to put the function here or not (keep in the tool). Usually the developer can come up with such a decision as work progress.
- If decision is made, move the function code here first and go through trial period.
- Interface can be still modified and updated as necessary since it's still private to public.
- When other tool(s) also use the function, the function can be moved to public common lib. (via discussion?)
- <<Continue adding functions as work progresses>>

3.4 Naming scheme

- Use uniform function names
  - H5tools_xxx_xxx() - common lib functions depending on HDF5
  - tools_xxx_xxx() - common lib functions not depending on HDF5
- Use uniform file names
  - H5tools_xxx.c - common lib files depending on HDF5
  - tools_xxx.c - common lib files not depending on HDF5

4 Work details

- Investigate the current problems and estimate the work for better planning and privatization.

- Review current library functions to figure out which function(s) belong where.
- Figure out repeated code and functions to eliminate the duplications in library code
- Review current tool functions to figure out which functions(s) belong where.
- Figure out repeated code and functions to eliminate the duplications in tools code
- Improve code quality
    - Update incorrect Indentations
    - Improve algorithm
    - Divide a big function to small functions

5 Developing Procedures

- Step 1: Separate the current code according to the design layout.
- Step 2: Adding additional ideas on each section (Common functions depend on HDF5 API, Common functions not depend on HDF5 API, Internal common functions depend on HDF5 APIs)
    - Prioritize additional ideas
    - Developing idea by priority (ex: must done now, can be done slowly)
- Step 3-1: Improve library code quality within the new frame work layout
    - Prioritize the work
    - Improve code by priority (ex: must done now, can be done slowly)
- Step 3-2: Update tool code
    - Start to enhance each of the tools with the new frame work layout
    - Prioritize the work
    - Rewrite tools' code by priority

6 Additional ideas

6.1 Separate data processing and viewing results (various way)

- Visualization API/tool using output from current displaying tools. (ex: h5dump)
- Outputs can be vary (xml, binary, ascii, etc).
- By separating viewing from processing should be more effective to maintain code between processing and viewing.
- This suggestion is relate to 3.2.

6.2 XML configure script for all tools

- Can be used for giving user control on any fixed value in the code. (ex: FLT_EPSILON/DBL_EPSILON are predefined value)

- Can be used to define command parameter options.
- Can be used to control tools behavior

```
Example:

<hdf5_tools>
      <h5diff>
            <option>
                  <VERSION>
                  <parameter>V</parameter>
                  <default>OFF</default>
                  </VERSION>
            </option>
            <option>
                  <REPORT>
                  <parameter>r</parameter>
                  <default>OFF</default>
                  </REPORT>
            </option>
            <definition>
                  <name>FLT_EPSILON</name>
                  <value>1.19209E-07</value>
            </definition>
      </h5diff>
      <h5dump>
            <option>
                  <REGION_DISPLAY>
                  <parameter>R</parameter>
                  <default>OFF</default>
                  </REGION_DISPLAY>
            </option>
            <definition>
                  <name>width</name>
                  <value>80</value>
            </definition>
      </h5dump>
</hdf5_tools>
```

7 Considerations

- How to relate 'common functions depending on HDF5 APIs' with H5 Lite Library?
- Internal common functions formation?
  - Keep this in separate dir and build tools at compile time.
  - Keep this in separate dir and make separate library to build tools at link time.
- Documentation for algorithm of each tools (flow-chart or UML)?

Acknowledgements

Thanks Peter and Elena for the initial input. Thanks Allen for additional idea.

Revision History

September 7, 2009:	Version 1 circulated for comment within The HDF Group.