Optional Feature Support in HDF5 Tools

Albert Cheng
The HDF Group

Problem Description

- Optional feature created for some HDF5 tools:
  - Special needs of individual user group;
  - Non public data encryption, compression, …
  - Privacy
  - Experimental
- How to maintain different optional features without "leaking" them to the official version of the tool

Criteria of Possible Solutions

- Cost of Implementation
  - Official HDF5 tools implementation cost
  - Optional feature implementation cost
- Cost of Maintenance
  - Official HDF5 tools maintenance cost
  - Optional feature maintenance cost
- Features supported:
  - Privacy of the optional feature
  - Lean size of the official HDF5 tool, both source and binary
  - Future extensions

Solution 1:Individual Source Branch

- A copy of the HDF5 source branch off the HDF5 official source branch; added optional features code to the copy.
- Implementation Cost:
  - HDF5: none.
  - Optional feature: initial implementation
- Maintenance Cost:
  - HDF5: none
  - Optional feature: high maintenance as it has to match changes in the HDF5 tool source

Solution 1 (cont.)

- Features supported:
  - Privacy maintained.
  - Lean size of the official HDF5 tool, both source and binary
  - Future extensions: very hard

Solution 2:Individual Source Branch

- Optional feature code are inserted into the HDF5 tool code but is compiled in only if it is configured in.
- Implementation Cost:
  - HDF5: initial implementation.
  - Optional feature: initial implementation
- Maintenance Cost:
  - HDF5: high; shots up when more optional features are added
  - Optional feature: lower since there is better coordination between the HDF5 and optional feature code

Solution 2 (cont.)

- Features supported:
  - No privacy since HDF5 source is public.
  - HDF5 tools size:
    - Source: swell up
    - Binary: lean
  - Future extensions: hard

Solution 3: External Library

- Functional interface of optional feature defined
- HDF5 tool modified to call the optional feature if configured in
- Optional Feature coded as an external library similar to szip library calls
- Implementation Cost:
  - HDF5: higher
  - Optional feature: initial implementation
- Maintenance Cost:
  - HDF5: none
  - Optional feature: lower

Solution 3 (cont.)

- Features supported:
  - Privacy maintained.
  - Lean size of the official HDF5 tool, both source and binary
  - Future extensions: possible

Solution 3 H5dump Quality Flag Example

- Functions of h5dump
  - Command line option parser
  - Dataset Rawdata Read (e.g., data transformation)
  - Rawdata display (e.g., scale calibration, quality flag masking)

Solution 4: Registered Pluggins

- Design and define HDF5 tool functionalities as function module
- Optional feature is coded according to the module definitions
- HDF5 tool coded to register each module according to build-time configure and/or runtime module registration

Solution 4: (cont.)

- Implementation Cost:
  - HDF5: High design cost.
  - Optional feature: initial implementation
- Maintenance Cost:
  - HDF5: Low
  - Optional feature: Low
- Features supported:
  - Privacy maintained.
  - Lean size of the official HDF5 tool, both source and binary
  - Future extensions: very extendible

Solution 4 H5dump Functions Example

- Functions of h5dump
  - Command line option parser
  - Dataset access
    - Dataset Open (e.g., encryption key, remote access)
    - Dataset Metadata Read (e.g., attribute “unit”)
    - Dataset Rawdata Read (e.g., data transformation)
    - Dataset Close (e.g., file lock release)
  - Display
    - Metadata display
    - Rawdata display (e.g., scale calibration, quality flag masking)

Summary of costs

Solution	Implementation Costs	Maintenance Costs	Other advantages
Duplicated copy	Low	Highest	Privacy retained; not extendible
Merged	Higher	Highest	Privacy lost; not extendible
External library	Higher	Lower	Privacy retained; Extendible
Registered Pluggin	Highest	Lowest	Privacy retained; Very extendible

Questions?