# Developer's Guide to Supported Roccom File Formats

**J. Norris**

**12/14/06**

## *1. Introduction*

The Roccom modules Rocin and Rocout are used to read and write data files in any one of the supported formats (currently HDF4 and CGNS). In addition, the main visualization tool, Rocketeer, can read Roccom-compliant data files. Developers have adopted certain conventions to overcome various limitations inherent in each of these formats. The purpose of this document is to describe in detail these conventions. Developers should already be familiar with the general layout and API of the base file format for HDF4 and CGNS files.

## *2. HDF4*

HDF4 is the oldest and most thoroughly exercised file format. Unfortunately, the "scientific dataset" interface does not directly support grid types more complicated than rectilinear, nor does it support multiblock grids. We are able to store data for these types of meshes despite this.

### 2.1. Dataset Organization

Data is separated into "blocks." In this document, a block refers to all of the data from a single grid section (or pane) of a given material (or window) for a single time. The data for a block is contiguous, although there is support for storing geometry and topology in a separate file. A file may contain data for multiple blocks and/or multiple times.

#### 2.1.1. Block Header

The first dataset for a block is called the block header. Its purpose is to mark the beginning of a new block of data, and to provide information applicable to the entire block. The dataset must be a 1D array of type char (DFNT_CHAR8). The predefined data attribute strings must hold the following values:

| | |
|---|---|
| `label` | A unique identifier for the mesh block, e.g. "block_001" or "1023". This generally corresponds to the Roccom pane id. |
| `units` | The time level, e.g. "10", or the actual time, e.g. "7.1e-05". |
| `format` | The string "block header". |
| `coordsys` | The material name, e.g. "gas". This generally corresponds to the Roccom window name. |

The data array itself is a string containing the mesh type, ghost information, and optionally, the path to the file containing nodal coordinates and connectivity information. The mesh type is represented as a number:

| | |
|---|---|
| 1 | Rectilinear |
| 2 | Structured |
| 3 | Unstructured |
| 4 | Discontinuous Galerkin |
| 5 | Polygonal (2D elements) |

The ghost information is a comma-separated list of numbers. If there is only one number, then this represents the number of ghost layers (structured meshes) or the number of ghost elements (unstructured meshes). If there is more than one number, then the first number is the number of ghost nodes, and the remaining numbers are the number of ghost elements in each connectivity table. This form is only used for the unstructured mesh types. The mesh type and ghost information are separated by a '|' character. If there is a path to a geometry file, then this is also separated by a '|'. For example:

"2|1" signifies a structured mesh with one ghost layer.

"3|25,0,0,35" signifies an unstructured mesh with 25 ghost nodes and 35 ghost elements at the end of the third connectivity table.

"5|0|mesh.hdf" signifies a polygonal mesh with zero ghost nodes and zero ghost elements, and an external geometry file named "mesh.hdf".

Note: Originally, the ghost information was also meant to report the number of connectivity tables for unstructured-type grids. A list of numbers of length n indicated a block with n-1 connectivity tables (min 1). However, due to various Roccom issues involving registering data, the ghost information provided in the block header may not be accurate if it's a single number. For this reason, when reading HDF4 files, a program should manually search for connectivity tables, and parse the labels of the nodal coordinate and connectivity table datasets to determine conclusively the number of ghost nodes and elements.

### 2.1.2. Nodal Coordinates

The datasets for the nodal coordinates either appear immediately after the block header if they are not stored in a separate file. They can be either float (DFNT_FLOAT32) or double (DFNT_FLOAT64). Their rank and size depend on the mesh type, as well as the number of nodes. For rectilinear meshes, the datasets should be 1D, and their length is equal topological dimension in the corresponding direction. A 5x7x9 rectilinear block would have an x coordinate dataset of length 5, a y coordinate dataset of length 7, and a z coordinate dataset of length 9. Structured meshes have 3D nodal coordinates with the same dimensions as the mesh. The various unstructured-type meshes have 1D nodal coordinate datasets a length equal to the number of nodes in the block.

In addition to the actual nodal coordinate data, meta-data is stored in the four predefined attribute strings:

| | |
|---|---|
| `label` | This follows the general format for labels, using "nc" as the attribute name. |
| `units` | The unit of measurement, usually "m". |
| `format` | This attribute should be an empty string. |
| `coordsys` | The material name and the block name, separated by a '\|'. This is used to locate the correct geometry data when it's stored in a separate file. |

The minimum and maximum "non-ghost" values of each component are stored and retrieved using the appropriate "getrange" or "setrange" function.

### 2.1.3. Connectivity Tables

Unstructured-type grids usually have connectivity tables to specify topology (a grid that does not contain any cells will not have a connectivity table). These tables are found immediately after the nodal coordinates. Connectivity tables are 2D arrays of integers (DFNT_INT32). The x dimension of the array is equal to the number of elements in the table. The y dimension should be equal to the number of nodes in the elements of this table (e.g. 3 for triangles, 8 for hexahedrons). The four predefined attribute strings also contain important information:

| | |
|---|---|
| `label` | This follows the general format for labels. The attribute name is based on the element type, and must start with a ':', e.g. ":t3" or ":B20:virtual". See the Roccom user guide for details. |
| `units` | The unit of measurement, usually "m". |
| `format` | This attribute should be the Roccom attribute name. |
| `coordsys` | The material name and the block name, separated by a '\|'. This is used to locate the correct geometry data when it's stored in a separate file. |

### 2.1.4. Variable Data

Window, panel, elemental and nodal Roccom attributes other than the nodal coordinates and connectivity tables are found immediately after the geometry datasets (or the block header dataset, if an external geometry file is used). Attributes may appear in any order. The components of multi-component attributes (e.g. vectors and tensors) are stored separately, but contiguously and in order. The data may be of any type. Datasets for window and panel attributes may be of any size or dimension. Datasets for elemental and nodal attributes are always 3D for structured-type grids and 1D for unstructured-type grids.

The four predefined attribute strings also contain important information:

| | |
|---|---|
| `label` | This follows the general format for labels. |
| `units` | The unit of measurement, if any. |
| `format` | The variable name, e.g. “temperature” or “velocity”. |
| `coordsys` | This attribute specifies which component this dataset contains.<br>0 – scalar (just one component)<br>1, 2 or 3 – vector component<br>11, 12, 13, 21, …, 33 – tensor component. |

The minimum and maximum “non-ghost” values of each component are stored and retrieved using the appropriate “getrange” or “setrange” function.

## 2.2. Format of Dataset Labels

The predefined HDF4 attribute “label” is used to specify a great deal of information about a Roccom attribute, including nodal coordinates and connectivity tables. The format for the label of a non-NULL, nonempty Roccom scalar attribute is as follows:

“name|x,G AT TIME XXXXXXXX”

where “name” is the Roccom name, “x” is ‘w’, ‘p’, ‘c’, ‘e’ or ‘n’ (for window, panel, conn, elemental or nodal attributes), “G” is the number of ghost items, and “XXXXXXXX” is the time. If the dataset is storing a component of a multi-component attribute, then the component number is placed at the front of the string, separated from the name by a hyphen. If the attribute is empty (size 0), then the comma is replaced with a 0. If the attribute is nonempty but NULL, then the comma is replaced by an @ symbol.

Examples:

| | |
|---|---|
| `temperature|e,5 AT TIME …` | Elemental attribute “temperature” with 5 ghost items. |
| `3-nc|n,2 AT TIME …` | Third component of nodal attribute “nc” (z coordinates) with 2 ghost items. |
| `:q4|p,0 AT TIME …` | Panel attribute “:t3” (a connectivity table) with no ghost items. |
| `pressure|e00 AT TIME …` | Empty elemental attribute “pressure”. |
| `turb|n@0 AT TIME …` | Null nodal attribute “turb”. |

## 2.3. Empty Datasets

The HDF file format does not allow datasets of zero length. Therefore empty and NULL datasets must contain a single dummy value. Be sure to parse the dataset label in order to differentiate an empty or NULL dataset from a dataset of length 1.

# 3. CGNS

Roccom was recently modified to support the CGNS format. Unlike the older HDF4 format, CGNS does support both structured and unstructured grid types, as well as multiblock grids. This file format works will for our purposes, but it does have a few limitations.

## 3.1. CGNS Hierarchy

Data in a CGNS file is stored in a tree hierarchy. The root node is a "Base" node, which corresponds nicely to a Roccom window (or material). Other types of nodes are stored under a `Base_t` node, including `Zone_t` nodes, which correspond to Roccom panes. Under the `Zone_t` nodes are `GridCoordinates_t`, `Elements_t` and `FlowSolution_t` nodes, which we use to store nodal coordinates, connectivity tables, elemental attribute data, and nodal attribute data, respectively. `IntegralData_t` nodes are used to store window, panel and conn attribute data. These nodes can be found under both the `Base_t` node (for window attribute data) and `Zone_t` nodes (for panel attribute data). Ghost information is stored using `Rind_t` nodes (for elemental and nodal data) or `Descriptor_t` nodes (for window, panel and conn data). Unit information is stored using `DimensionalUnits_t` and `DimensionalExponents_t` nodes, as well as `Descriptor_t` nodes. Time information is stored in the `BaseIterativeData_t` and `ZoneIterativeData_t` nodes. Data range information is stored in `Descriptor_t` nodes under each `DataArray_t` node. Care is taken to insure SIDS compliance, allowing the use of 3rd party software such as Tecplot to analyze data. Please review the SIDS document for a full description of the CGNS tree structure and node types.

## 3.2. Node Naming Conventions

Roccom uses the following rules to generate nodes names for the various CGNS nodes.

| **Node type** | **Node name** |
|---|---|
| `Base_t` | The window (material) name |
| `Zone_t` | The pane id (four digits, padded with leading zeroes) |
| `GridCoordinates_t` | "Grid" + the time level string |
| `Elements_t` | The Roccom connectivity table name. Empty tables prepend "Empty" to the name. NULL tables prepend "Null". |
| `IntegralData_t` | "WinData" + the time level string (for window attributes)<br>"PaneData" + the time level string (for pane attributes)<br>"ConnData" + the time level string (for conn attributes) |
| `FlowSolution_t` | "ElemData" + the time level string<br>"NodeData" + the time level string |
| `Descriptor_t` | "Range" (for the data range of a single component)<br>"MagnitudeRange" (for the range of a vector's magnitude)<br>"TraceRange" (for the range of a tensor's trace)<br>"Units" (for unit of measurement as a string)<br>"Ghost" (number of ghost items of a window, pane or conn attribute) |

Note that the SIDS standard requires that the first `GridCoordinates_t` node be named "GridCoordinates". When writing the first `GridCoordinates_t` node for a zone, name it "GridCoordinates", but then create a link to it using the conventional name described above.
Node names are limited to 32 characters. In the event that a generated name exceeds the maximum length, the prefix (e.g. "ElemData") is shortened to reduce the final length to 32.

## 3.3. Nonstandard Node Use

- Every `DataArray_t` node should have an underlying `Descriptor_t` node named "Range". Its value string is usually of the form "min, max", and should report the minimum and maximum non-ghost value in the array. For empty arrays, the value of the "Range" descriptor should be "EMPTY". For NULL arrays, the value should be "NULL".
- "MagnitudeRange" and "TraceRange" descriptors are stored under the `DataArray_t` node of the first component of the vector or tensor.
- The "Units" descriptor, while technically redundant, preserves the unit string from Roccom. The `DimensionalUnits_t` and `DimensionalExponents_t` nodes only store unit information in terms of the most basic units of mass, length, time, temperature, and angle measurement, so a unit such as joule is transformed into $kg{\cdot}m^2 / s^2$.
- A "Ghost" descriptor is used under `IntegralData_t` nodes because `Rind_t` nodes cannot be placed under `IntegralData_t` or `DataArray_t` nodes. `Rind_t` nodes are used as normal for elemental and nodal attribute data, as well as connectivity tables.
- A `FlowSolution_t` node can contain either node-centered data or element-centered data, but not both. For this reason we must use two `FlowSolution_t` nodes per timestep.
- The `BaseIterativeData_t` and `ZoneIterativeData_t` nodes provide pointers to only the `GridCoordinates_t` and node-centered `FlowSolution_t` nodes for each time dump. This necessitates the naming convention used for attribute data.

## 3.4. Special Handling of Panel Attributes

Some special panel attributes are processed differently when writing a CGNS file.

### 3.4.1. "ridges"

If the "ridges" attribute exists, it is written out as normal, but in addition a second base and unstructured zone are created in the file. The new base has the same name as the original concatenated with "_ridges". The zone name is the same as the original, although the type changes to unstructured. The `GridCoordinates_t` node and `FlowSolution_t` node are links to the original `GridCoordinates_t` and node-centered `FlowSolution_t` nodes. The `Elements_t` node is an array of line elements whose node ids are given in the "ridges" attribute.

#### 3.4.1. “pconn” and “bc”
Although there is no special handling for these attributes at present, CGNS offers special nodes to describe zone connectivity and boundary conditions.

## 3.4. Empty `DataArray_t` Nodes
The CGNS file format does not allow arrays of zero length.  In addition, DataArray_t nodes stored under a FlowSolution_t node must be the correct size.  Therefore empty and NULL arrays must contain dummy values.  Be sure to check the “Range” descriptor in order to identify empty or NULL arrays.